

# **Positionsbasierte Routingverfahren und kooperative Anwendungen für drahtlose Ad-hoc-Netzwerke**

Inauguraldissertation  
zur Erlangung des akademischen Grades  
eines Doktors der Naturwissenschaften  
der Universität Mannheim

vorgelegt von  
Dipl. Wirtsch.-Inf. Sascha Schnauffer  
aus Heidelberg

Mannheim, 2014

Dekan: Prof. Dr. Heinz Jürgen Müller, Universität Mannheim  
Referent: Prof. Dr. Wolfgang Effelsberg, Universität Mannheim  
Korreferent: Prof. Dr. Martin Mauve, Heinrich-Heine-Universität Düsseldorf

Tag der mündlichen Prüfung: 11.09.2014

# Zusammenfassung

Die Verbreitung von leistungsfähigen portablen Geräten – wie zum Beispiel Smartphones, Tablet-Computern oder Laptops – hat in den letzten Jahren stark zugenommen. Allerdings werden die verbauten Funktechnologien fast ausschließlich für den direkten Zugriff auf die bestehende Infrastruktur verwendet. So wird etwa das üblicherweise verbaute WLAN-Modul primär für den Internetzugriff eingesetzt. Prinzipiell können derartige Geräte aber auch ein drahtloses Ad-hoc-Netzwerk aufbauen, in welchem alle Daten direkt zwischen den Geräten ausgetauscht werden.

Damit in einem solchen Netzwerk auch nicht benachbarte Geräte miteinander kommunizieren können, wird ein geeignetes Routingprotokoll benötigt. Anstatt die Topologie des Netzwerks durch Fluten zu erkunden, kann ein Routingprotokoll auch auf die geographischen Positionsinformationen der Netzwerkteilnehmer zurückgreifen. Da diese Strategie insbesondere in hoch mobilen Umgebungen vielversprechend ist, werden in dieser Arbeit mehrere positionsbasierte Routingprotokolle detailliert analysiert und mit Hilfe einer umfangreichen Simulationsstudie untersucht. Hierfür wird ein neuartiger Simulator verwendet, der die großflächige Simulation eines Innenstadtbereichs ermöglicht und dabei auch die zu erwartenden Funkhindernisse berücksichtigt. Des Weiteren wird mit *Greedy Routing with Abstract Neighbour Table* ein speziell für die beschriebene Umgebung optimiertes Routingprotokoll vorgestellt.

Während des Routings wird der Paketinhalt nicht von weiterleitenden Knoten verwendet. Die Kommunikation erfolgt auf der Anwendungsebene daher nur zwischen dem Start- und dem Zielgerät. Sollen allerdings alle Geräte gemeinsam eine bestimmte Ausgabe erfüllen, so sind die weitergeleiteten Daten meist für alle Geräte von Interesse. Daher wird in der vorliegenden Arbeit die genannte Kommunikationsform exemplarisch anhand der dezentralen Aggregation untersucht. Ein vielversprechender Ansatz basiert auf einer probabilistischen Datenstruktur, die aus diesem Grund detailliert analysiert wird. Dabei werden sowohl die Leistungsfähigkeit der Datenstruktur an sich als auch die resultierende Genauigkeit der dezentralen Aggregation untersucht. Des Weiteren werden mit *Bloomfilter-Maps* und *Bloomfilter-Arrays* zwei neue Datenstrukturen vorgestellt, welche die erreichbare Genauigkeit signifikant verbessern.

Im letzten Teil der Arbeit wird analysiert, wie die präsentierten Verfahren am besten in die bestehende IT-Infrastruktur integriert werden können und welche Funktechnologie dabei verwendet werden sollte. In diesem Zusammenhang wird ein Mechanismus zum Austausch von Kurznachrichten zwischen nicht assoziierten WLAN-Geräten vorgeschlagen, der *Beacon-Based Short Message Exchange* genannt wird. Ebenfalls werden der entstandene Prototyp und die angedachte Systemarchitektur präsentiert. Da der BSMX-Mechanismus auch für den Austausch von Nachrichten zwischen Access Points verwendet werden kann, werden in dieser Arbeit die Ergebnisse einer großflächigen Messung in der Mannheimer Innenstadt vorgestellt. Die Messdaten werden des Weiteren für eine Simulation der Informationsverbreitung innerhalb eines Netzwerks von uneinheitlich konfigurierten Access Points verwendet.





# Abstract

The use of high-performance portable devices such as smartphones, tablet computers and laptops has increased tremendously in recent years. However, the wireless technologies installed in such devices are almost always exclusively used for directly accessing existing infrastructures. As a result, the WLAN module in the standard installation is primarily used for accessing the Internet. In principle, devices of this type are also capable of creating a wireless ad-hoc network so that all data can be directly shared between devices.

A suitable routing protocol is needed so that non-adjacent devices in such a network are able to communicate with one another. Instead of discovering the network topology through flooding, a routing protocol is also capable of using the positional information of network participants. This strategy is particularly promising in the context of ultra-mobile environments, which is why this dissertation will conduct a detailed analysis of multiple position-based routing protocols and explore them on the basis of an extensive simulation study. This will be done using an innovative simulator, which enables the simulation of large city center areas and also accounts for radio barriers in the process. Furthermore, *Greedy Routing with Abstract Neighbour Table*, a routing protocol specially optimised for such environments, will also be presented.

During the routing process, the packet content is not used by forwarding nodes. This is why the communication at the application level only takes place between the source and target device. If, however, all devices are collectively completing a certain task, then the data forwarded is most likely of interest for all devices. This dissertation will therefore be exploring this form of communication using examples based on decentralised aggregation. A probabilistic data structure offers a promising approach and will be analysed in detail for this reason. In the process, both the performance of the data structure as well as the resulting accuracy of decentralised aggregation will be investigated. Furthermore, two new data structures, *Bloomfilter-Maps* and *Bloomfilter-Arrays*, will be presented; both of them significantly improve the attainable accuracy.

The last part of this dissertation will analyse the best way to integrate the methods presented herein in existing IT infrastructures, and which wireless technology should be used in this respect. *Beacon-Based Short Message Exchange*, a mechanism for exchanging short text messages between non-associated WLAN devices, will be proposed in this context. The resulting prototype and the considered system architecture will also be presented. As the BSMX mechanism can also be used for exchanging messages between access points, the results of a large-scale measurement in the city center of Mannheim will be introduced. The measurement data will also be used to simulate information distribution within a network comprising non-uniformly configured access points.



# Inhaltsverzeichnis

|   |             |
|---|-------------|
| <b>Abbildungsverzeichnis</b>  | <b>vii</b>  |
| <b>Tabellenverzeichnis</b>  | <b>xiii</b> |
| <b>Abkürzungsverzeichnis</b>  | <b>xv</b>   |
| <b>1 Einleitung</b>   | <b>1</b>    |
| <b>2 Grundlagen der drahtlosen Kommunikation in Ad-hoc-Netzwerken</b>           | <b>7</b>    |
| 2.1 Ausgewählte Routingverfahren . . . . .                                      | 9           |
| 2.1.1 Ad hoc On-Demand Distance Vector Routing . . . . .                        | 11          |
| 2.1.2 Positionsbasierte Greedy-Algorithmen . . . . .                            | 14          |
| 2.1.3 Face-2 Routing . . . . .  | 19          |
| 2.1.4 Kombinationsmöglichkeiten verschiedener Verfahren . . . . .               | 22          |
| 2.1.5 Verfahren zur Wegewahl mit globalem Zusatzwissen . . . . .                | 25          |
| 2.2 Kooperative Anwendungen . . . . .   | 26          |
| 2.2.1 Anwendungen in Sensornetzwerken . . . . .                                 | 27          |
| 2.2.2 Anwendungen in Fahrzeug-ad-hoc-Netzwerken . . . . .                       | 28          |
| 2.3 Zusammenfassung . . . . .   | 33          |
| <b>3 Positionsbasierte Routingverfahren für den Einsatz in Innenstädten</b>     | <b>35</b>   |
| 3.1 Verallgemeinerung bestehender positionsbasierter Routingverfahren . . . . . | 36          |
| 3.2 Erstellen einer geeigneten Simulationsumgebung . . . . .                    | 40          |
| 3.2.1 Netzwerktopologie und Funkhindernisse . . . . .                           | 46          |
| 3.2.2 Funktionsweise und Konfiguration des Simulators . . . . .                 | 50          |
| 3.2.3 Definition der Bewertungskriterien . . . . .                              | 53          |
| 3.3 Simulationsergebnisse bestehender Verfahren . . . . .                       | 54          |
| 3.3.1 Knotendichte und Knotenverteilung . . . . .                               | 54          |
| 3.3.2 Greedy-Algorithmen . . . . .  | 58          |
| 3.3.3 Greedy-Algorithmen mit Überbrückungsverfahren . . . . .                   | 62          |
| 3.3.4 Greedy-Algorithmen mit Kreuzungserkennung . . . . .                       | 68          |
| 3.4 Greedy Routing with Abstract Neighbour Table . . . . .                      | 74          |
| 3.4.1 Theoretische Vergleichsverfahren . . . . .                                | 74          |
| 3.4.2 Abstract Neighbour Table . . . . .  | 78          |
| 3.4.3 Die Metrik von Greedy Routing with Abstract Neighbour Table . . . . .     | 81          |
| 3.4.4 Simulationsergebnisse . . . . .   | 84          |
| 3.4.5 Abschätzen der hindernisfreien Umgebung . . . . .                         | 87          |
| 3.5 Vergleich der Überbrückungsverfahren . . . . .                              | 88          |
| 3.6 Zusammenfassung . . . . .   | 110         |

|          |  |            |
|----------|--|------------|
| <b>4</b> | <b>Probabilistische Datenstrukturen für kooperative Anwendungen</b>            | <b>113</b> |
| 4.1      | Flajolet-Martin Sketches   | 116        |
| 4.1.1    | Aufbau und Eigenschaften   | 116        |
| 4.1.2    | Relativer Fehler und Standardabweichung  | 118        |
| 4.1.3    | Weiterentwicklungen  | 125        |
| 4.1.4    | Verbesserung der Schätzung   | 126        |
| 4.2      | Kooperative Aggregation von orts- und zeitbezogenen Daten                      | 132        |
| 4.2.1    | Soft State Sketches  | 136        |
| 4.2.2    | Kompression  | 138        |
| 4.2.3    | Simulationsumgebung  | 143        |
| 4.2.4    | Simulationsergebnisse  | 145        |
| 4.3      | Bloomfilter-Maps   | 150        |
| 4.3.1    | Verlustfreie Datenübertragung  | 151        |
| 4.3.2    | Aufbau und Eigenschaften   | 153        |
| 4.3.3    | Fehlerabschätzung  | 156        |
| 4.3.4    | Simulationsergebnisse  | 159        |
| 4.4      | Bloomfilter-Arrays   | 163        |
| 4.4.1    | Datenstruktur und Schätzfunktion   | 164        |
| 4.4.2    | Simulationsergebnisse  | 165        |
| 4.5      | Zusammenfassung  | 169        |
| <b>5</b> | <b>Integration kooperativer Anwendungen in die bestehende IT-Infrastruktur</b> | <b>171</b> |
| 5.1      | Etablierte Funktechnologien in mobilen Geräten                                 | 171        |
| 5.2      | Ad-hoc-Netzwerke mit IEEE 802.11 WLAN  | 174        |
| 5.3      | Beacon-Based Short Message Exchange  | 177        |
| 5.3.1    | Technische Konzeption  | 178        |
| 5.3.2    | Architektur und Implementierung  | 180        |
| 5.3.3    | Beispielanwendungen  | 182        |
| 5.4      | Empirische Bestimmung der zu erwartenden Konnektivität                         | 184        |
| 5.4.1    | Versuchsaufbau und Durchführung  | 185        |
| 5.4.2    | Abschätzung der Empfangswahrscheinlichkeiten                                   | 186        |
| 5.4.3    | Konstruktion des Netzwerkgraphen   | 189        |
| 5.4.4    | Simulation der Informationsverbreitung   | 191        |
| 5.5      | Zusammenfassung  | 194        |
| <b>6</b> | <b>Zusammenfassung und Ausblick</b>  | <b>197</b> |
|          | <b>Literaturverzeichnis</b>  | <b>205</b> |

# Abbildungsverzeichnis

|     |  |    |
|-----|--|----|
| 2.1 | Kommunikationsmöglichkeiten innerhalb eines Unit Disk Graph . . . . .  | 8  |
| 2.2 | AODV: Die Pfadsuche zum Zielknoten . . . . .   | 11 |
| 2.3 | AODV: Der Rückweg zum Startknoten . . . . .  | 12 |
| 2.4 | Auswahl des am besten geeigneten Weiterleiters . . . . .   | 15 |
| 2.5 | Lokales Optimum von Greedy-Algorithmen . . . . .   | 16 |
| 2.6 | Delaunay-Triangulation . . . . .   | 17 |
| 2.7 | Gabriel Graph und Relative Neighbourhood Graph . . . . .   | 20 |
| 2.8 | Face-2: Die Pfadsuche zum Zielknoten . . . . .   | 20 |
| 2.9 | Überbrückung des lokalen Optimums von Greedy-Algorithmen . . . . .   | 23 |
| 3.1 | Schematische Darstellung von positionsbasiertem Routing (Übersicht) . . . . .  | 38 |
| 3.2 | Schematische Darstellung von positionsbasiertem Routing (Fortsetzung) . . . . .  | 39 |
| 3.3 | Screenshot der Webanwendung MapEdit . . . . .  | 49 |
| 3.4 | Simulationsgebiet mit Hindernislinien . . . . .  | 51 |
| 3.5 | Compass, GRS, MFR:<br>X-Achse: Gefundene Pfade [%]<br>Y-Achse: Funkreichweite [m] und Knotendichte [Knoten/m <sup>2</sup> ]<br>Konfig.: Funkreichweite: 100-500 m, Knotendichte: 10-70 k/m <sup>2</sup> . . . . .<br>Szenario: Knoten gleichmäßig in der <u>Ebene</u> verteilt, <u>keine</u> Funkhindernisse | 55 |
| 3.6 | Compass, GRS, MFR:<br>X-Achse: Gefundene Pfade [%]<br>Y-Achse: Knoten innerhalb der Funkreichweite [k]<br>Konfig.: Funkreichweite: 100-500 m, Knotendichte: 10-70 k/m <sup>2</sup> . . . . .<br>Szenario: Knoten gleichmäßig in der <u>Ebene</u> verteilt, <u>keine</u> Funkhindernisse                      | 56 |
| 3.7 | Compass, Compass-GG, GRS, MFR:<br>X-Achse: Anzahl der gefundenen Pfade [Millionen]<br>Y-Achse: Kommunikationsdistanz [m]<br>Konfig.: Funkreichweite: 100 m, 300 m, 500 m, Knotendichte: 25,37 k/km . . . . .<br>Szenario: Knoten gleichmäßig auf den Straßen verteilt, <u>keine</u> Funkhindernisse          | 57 |
| 3.8 | Compass, Compass-GG, GRS, MFR:<br>X-Achse: Gefundene Pfade [%]<br>Y-Achse: Funkreichweite [m]<br>Konfig.: Funkreichweite: 100-800 m, Knotendichte: 12,685 k/km, 25,37 k/km . . . . .<br>Szenario: Knoten gleichmäßig auf den Straßen verteilt, mit Funkhindernissen  | 59 |
| 3.9 | Compass, Compass-GG, GRS, MFR:<br>X-Achse: Anzahl der gefundenen Pfade [Millionen] und gefundene Pfade [%]<br>Y-Achse: Kommunikationsdistanz [m]<br>Konfig.: Funkreichweite: 300 m, Knotendichte: 25,37 k/km . . . . .<br>Szenario: Knoten gleichmäßig auf den Straßen verteilt, mit Funkhindernissen        | 61 |

|      |  |    |
|------|--|----|
| 3.10 | Compass, Compass-GG, GRS, MFR:   |    |
|      | X-Achse: Länge der gefundenen Pfade [Hops, m]                                    |    |
|      | Y-Achse: Kommunikationsdistanz [Hops, m]   |    |
|      | Konfig.: Funkreichweite: 300 m, Knotendichte: 25,37 k/km                         | 62 |
|      | Szenario: Knoten gleichmäßig auf den Straßen verteilt, mit Funkhindernissen      |    |
| 3.11 | Pfadsuche von GRS mit Face-2 als Überbrückungsstrategie                          | 63 |
| 3.12 | GRS und MFR mit DV- und Face-2-Überbrückung:                                     |    |
|      | X-Achse: Anzahl der gefundenen Pfade [Millionen] und gefundene Pfade [%]         |    |
|      | Y-Achse: Kommunikationsdistanz [m]   |    |
|      | Konfig.: Funkreichweite: 300 m, Knotendichte: 25,37 k/km                         | 64 |
|      | Szenario: Knoten gleichmäßig auf den Straßen verteilt, mit Funkhindernissen      |    |
| 3.13 | Gabriel Graph im Innenstadtszenario mit Funkhindernissen                         | 65 |
| 3.14 | Routingschleife von Face-2   | 66 |
| 3.15 | GRS und MFR mit DV- und Face-2-Überbrückung:                                     |    |
|      | X-Achse: Pfadlänge [Hops]  |    |
|      | Y-Achse: Kommunikationsdistanz [Hops]  |    |
|      | Konfig.: Funkreichweite: 300 m, Knotendichte: 25,37 k/km                         | 67 |
|      | Szenario: Knoten gleichmäßig auf den Straßen verteilt, mit Funkhindernissen      |    |
| 3.16 | Ergebnisse der Kreuzungserkennung:   |    |
|      | X-Achse: Korrekt erkannte Knoten [Prozentsatz der Knoten]                        |    |
|      | Y-Achse: Schwellenwert der Kreuzungserkennung                                    |    |
|      | Konfig.: Funkreichweite: 100-500 m, Knotendichte: 12,685 k/km, 25,37 k/km        | 70 |
|      | Szenario: Knoten gleichmäßig auf den Straßen verteilt, mit Funkhindernissen      |    |
| 3.17 | Pfadsuche von GRS mit und ohne Kreuzungserkennung                                | 71 |
| 3.18 | GRS mit Kreuzungserkennung und Face-2-Überbrückung:                              |    |
|      | X-Achse: Anzahl der gefundenen Pfade [Millionen] und gefundene Pfade [%]         |    |
|      | Y-Achse: Kommunikationsdistanz [m]   |    |
|      | Konfig.: Funkreichweite: 300 m, Knotendichte: 25,37 k/km                         | 72 |
|      | Szenario: Knoten gleichmäßig auf den Straßen verteilt, mit Funkhindernissen      |    |
| 3.19 | GRS mit Kreuzungserkennung sowie mit DV- und Face-2-Überbrückung:                |    |
|      | X-Achse: Pfadlänge [Hops]  |    |
|      | Y-Achse: Kommunikationsdistanz [Hops]  |    |
|      | Konfig.: Funkreichweite: 300 m, Knotendichte: 25,37 k/km                         | 73 |
|      | Szenario: Knoten gleichmäßig auf den Straßen verteilt, mit Funkhindernissen      |    |
| 3.20 | OptimalGreedy, GRS-xHop, GRS-JD, GRS:  |    |
|      | X-Achse: Gefundene Pfade [%]   |    |
|      | Y-Achse: Kommunikationsdistanz [Hops, m]   |    |
|      | Konfig.: Funkreichweite: 300 m, Knotendichte: 25,37 k/km                         | 77 |
|      | Szenario: Knoten gleichmäßig auf den Straßen verteilt, mit Funkhindernissen      |    |
| 3.21 | Durchschnittliche Größe einer Multi-Hop-Nachbarschaftstabelle:                   |    |
|      | X-Achse: Anzahl der Nachbarn und Anzahl der Nachbarn [%]                         |    |
|      | Y-Achse: Knotendichte [k innerhalb Funkreichweite]                               |    |
|      | Konfig.: Funkreichweite: 300 m, Knotendichte: 2-120 [k innerhalb Funkreichweite] | 79 |
|      | Szenario: Knoten gleichmäßig in der Ebene verteilt, keine Funkhindernisse        |    |
| 3.22 | Zwei Varianten der Abstract Neighbour Table (ANT)                                | 81 |
| 3.23 | Pfadsuche von GRANT und GRS mit Face-2-Überbrückung                              | 82 |
| 3.24 | Pfadsuche von GRANT mit Face-2-Überbrückung                                      | 84 |

|  |               |
|--|---------------|
| 3.25 GRANT mit Face-2-Überbrückung im Vergleich zu GRS-2Hop:                 |               |
| X-Achse: Anzahl der gefundenen Pfade [Millionen] und gefundene Pfade [%]     |               |
| Y-Achse: Kommunikationsdistanz [m]   |               |
| Konfig.: Funkreichweite: 300 m, Knotendichte: 25,37 k/km                     | · · · · · 86  |
| Szenario: Knoten gleichmäßig auf den Straßen verteilt, mit Funkhindernissen  |               |
| 3.26 GRANT mit Face-2-Überbrückung im Vergleich zu GRS-2Hop:                 |               |
| X-Achse: Pfadlänge [Hops] (vollständige und unvollständige Pfade)            |               |
| Y-Achse: Kommunikationsdistanz [Hops]  |               |
| Konfig.: Funkreichweite: 300 m, Knotendichte: 25,37 k/km                     | · · · · · 87  |
| Szenario: Knoten gleichmäßig auf den Straßen verteilt, mit Funkhindernissen  |               |
| 3.27 Abschätzung der hindernisfreien Gebiete anhand empfangener ANTs         | · · · · · 89  |
| 3.28 GRANT, GRS, GRS-JD mit DV- und Face-2-Überbrückung:                     |               |
| X-Achse: Gefundene Pfade [%]   |               |
| Y-Achse: Funkreichweite [m]  |               |
| Konfig.: Funkreichweite: 100-800 m, Knotendichte: 12,685 k/km, 25,37 k/km    | · · · · · 90  |
| Szenario: Knoten gleichmäßig auf den Straßen verteilt, mit Funkhindernissen  |               |
| 3.29 GRANT, GRS-JD mit DV- und Face-2-Überbrückung:                          |               |
| X-Achse: Gefundene Pfade [%]   |               |
| Y-Achse: Maximale Länge einer Überbrückung [Hops]                            |               |
| Konfig.: Funkreichweite: 100 m, 300 m, Knotendichte: 12,685 k/km, 25,37 k/km | · · · · · 92  |
| Szenario: Knoten gleichmäßig auf den Straßen verteilt, mit Funkhindernissen  |               |
| 3.30 GRANT, GRS, GRS-JD mit DV- und Face-2-Überbrückung:                     |               |
| X-Achse: Überbrückungen pro gefundenem Pfad                                  |               |
| Y-Achse: Funkreichweite [m] und Kommunikationsdistanz [m]                    |               |
| Konfig.: Funkreichweite: 100-800 m, Knotendichte: 12,685 k/km, 25,37 k/km    | · · · · · 94  |
| Szenario: Knoten gleichmäßig auf den Straßen verteilt, mit Funkhindernissen  |               |
| 3.31 GRANT, GRS, GRS-JD mit DV- und Face-2-Überbrückung:                     |               |
| X-Achse: Hops pro Überbrückung   |               |
| Y-Achse: Funkreichweite [m] und Kommunikationsdistanz [m]                    |               |
| Konfig.: Funkreichweite: 100-800 m, Knotendichte: 12,685 k/km, 25,37 k/km    | · · · · · 95  |
| Szenario: Knoten gleichmäßig auf den Straßen verteilt, mit Funkhindernissen  |               |
| 3.32 GRANT, GRS-JD mit DV- und Face-2-Überbrückung:                          |               |
| X-Achse: Pro Überbrückung verschickte Pakete                                 |               |
| Y-Achse: Maximale Länge einer Überbrückung [Hops]                            |               |
| Konfig.: Funkreichweite: 100-400 m, Knotendichte: 12,685 k/km, 25,37 k/km    | · · · · · 97  |
| Szenario: Knoten gleichmäßig auf den Straßen verteilt, mit Funkhindernissen  |               |
| 3.33 GRANT, GRS-JD mit DV- und Face-2-Überbrückung:                          |               |
| X-Achse: Datenvolumen für ein 1000-Byte-Datenpaket [kB]                      |               |
| Y-Achse: Maximale Länge einer Überbrückung [Hops]                            |               |
| Konfig.: Funkreichweite: 100 m, 300 m, Knotendichte: 12,685 k/km, 25,37 k/km | · · · · · 100 |
| Szenario: Knoten gleichmäßig auf den Straßen verteilt, mit Funkhindernissen  |               |
| 3.34 GRANT, GRS-JD mit DV- und Face-2-Überbrückung:                          |               |
| X-Achse: Gefundene Pfade [%] und Datenvolumen für ein 1000-Byte-Paket [kB]   |               |
| Y-Achse: Kommunikationsdistanz [m]   |               |
| Konfig.: Funkreichweite: 300 m, Knotendichte: 12,685 k/km, 25,37 k/km        | · · · · · 101 |
| Szenario: Knoten gleichmäßig auf den Straßen verteilt, mit Funkhindernissen  |               |
| Sonstiges: Überbrückung auf 3 beziehungsweise 15 Hops limitiert              |               |

|      |   |     |
|------|---|-----|
| 3.35 | GRANT, GRS-JD mit DV- und Face-2-Überbrückung:                              |     |
|      | X-Achse: Datenvolumen [kB]  |     |
|      | Y-Achse: Nutzdaten [kB]   |     |
|      | Konfig.: Funkreichweite: 300 m, Knotendichte: 12,685 k/km, 25,37 k/km       | 103 |
|      | Szenario: Knoten gleichmäßig auf den Straßen verteilt, mit Funkhindernissen |     |
|      | Sonstiges: Überbrückung auf 3 bzw. 15 Hops limitiert                        |     |
| 4.1  | Kombination von FM-Sketches   | 118 |
| 4.2  | Verteilung der Hashwerte von Lookup3 und SHA-1                              | 121 |
| 4.3  | PCSA:   |     |
|      | X-Achse: Anzahl der FM-Sketches   |     |
|      | Y-Achse: Relativer Fehler [%] und Standardabweichung [%]                    | 122 |
|      | Konfig.: Synthetisch generierte Elemente                                    |     |
| 4.4  | PCSA mit modifizierter Abschätzung:   |     |
|      | X-Achse: Eingefügte Elemente  |     |
|      | Y-Achse: Relativer Fehler [%] und systematischer Fehler [%]                 | 123 |
|      | Konfig.: Synthetisch generierte Elemente, Anfang des Wertebereichs          |     |
| 4.5  | PCSA mit modifizierter Abschätzung:   |     |
|      | X-Achse: Eingefügte Elemente  |     |
|      | Y-Achse: Relativer Fehler [%] und systematischer Fehler [%]                 | 124 |
|      | Konfig.: Synthetisch generierte Elemente, gesamter Wertebereich             |     |
| 4.6  | Die relative Häufigkeit von Bitmustern in FM-Sketches:                      |     |
|      | X-Achse: Eingefügte Elemente  |     |
|      | Y-Achse: Relative Häufigkeit [%]  | 128 |
|      | Konfig.: Synthetisch generierte Elemente                                    |     |
| 4.7  | PCSA-LT:  |     |
|      | X-Achse: Anzahl der FM-Sketches   |     |
|      | Y-Achse: Relativer Fehler [%] und Standardabweichung [%]                    | 130 |
|      | Konfig.: Synthetisch generierte Elemente                                    |     |
| 4.8  | Dezentrale Aggregation auf der niedrigsten Hierarchieebene                  | 134 |
| 4.9  | Dezentrale Aggregation mit Hilfe von Soft State Sketches                    | 137 |
| 4.10 | PCSA und PCSA-LT (AVG, COUNT/SUM):  |     |
|      | X-Achse: Anzahl der Sketches  |     |
|      | Y-Achse: Relativer Fehler [%]   | 146 |
|      | Konfig.: Simulation im Innenstadtdgebiet mit zufälligen Sensorwerten        |     |
| 4.11 | PCSA und PCSA-LT (Nachbarn in einem bestimmten Hopradius):                  |     |
|      | X-Achse: Abstand [Hops]   |     |
|      | Y-Achse: Relativer Fehler [%] und absoluter Fehler [Knoten]                 | 148 |
|      | Konfig.: Simulation im Innenstadtdgebiet mit zufälligen Sensorwerten        |     |
| 4.12 | Benötigter Speicherplatz von Soft State Sketches:                           |     |
|      | X-Achse: Anzahl der Sketches  |     |
|      | Y-Achse: Größe [Byte]   | 149 |
|      | Konfig.: Simulation im Innenstadtdgebiet mit zufälligen Sensorwerten        |     |
| 4.13 | Entropie eines Bitarrays in Abhängigkeit von der Anzahl der gesetzten Bits: |     |
|      | X-Achse: Anzahl gesetzter Bits  |     |
|      | Y-Achse: Entropie pro gesetztem Bit [Bit]                                   | 154 |
| 4.14 | Exemplarischer Aufbau einer Bloomfilter-Map                                 | 155 |
| 4.15 | Bloomfilter-Map (COUNT):  |     |
|      | X-Achse: Eingefügte Elemente und eingefügte Elemente in Relation zu B [%]   |     |
|      | Y-Achse: Relativer Fehler [%] und Entropie in Relation zu B [%]             | 157 |
|      | Konfig.: Mathematische Abschätzung der Anzahl der gesetzten Bits            |     |



|  |     |
|--|-----|
| 4.16 Bloomfilter-Map (COUNT):  |     |
| X-Achse: Eingefügte Elemente in Relation zu B [%]  |     |
| Y-Achse: Relativer Fehler [%] und rel. Fehler in Relation zu rel. Fehler bei $x = 5\%$ [%] | 158 |
| Konfig.: Synthetisch generierte Elemente   |     |
| 4.17 Bloomfilter-Map (COUNT, SUM, AVG):  |     |
| X-Achse: Anzahl der TTL-Zähler   |     |
| Y-Achse: Relativer Fehler [%]  | 160 |
| Konfig.: Simulation im Innenstadtgebiet mit zufälligen Sensorwerten                        |     |
| 4.18 Bloomfilter-Map (Nachbarn in einem bestimmten Hopradius):                             |     |
| X-Achse: Abstand [Hops]  |     |
| Y-Achse: Relativer Fehler [%] und absoluter Fehler [Knoten]                                | 161 |
| Konfig.: Simulation im Innenstadtgebiet mit zufälligen Sensorwerten                        |     |
| 4.19 Benötigter Speicherplatz von Bloomfilter-Maps:  |     |
| X-Achse: Anzahl der TTL-Zähler   |     |
| Y-Achse: Größe [Byte] und Anzahl der gespeicherten Sensorwerte                             | 162 |
| Konfig.: Simulation im Innenstadtgebiet mit zufälligen Sensorwerten                        |     |
| 4.20 PCSA und Bloomfilter-Array:   |     |
| X-Achse: Anzahl der Sketches (M) bzw. Größe der Bloomfilter (B)                            |     |
| Y-Achse: Relativer Fehler [%] und Standardabweichung [%]                                   | 165 |
| Konfig.: Synthetisch generierte Elemente   |     |
| 4.21 PCSA und Bloomfilter-Array:   |     |
| X-Achse: Eingefügte Elemente   |     |
| Y-Achse: Relativer Fehler [%]  | 166 |
| Konfig.: Synthetisch generierte Elemente, Anfang des Wertebereichs                         |     |
| 4.22 PCSA und Bloomfilter-Array (COUNT):   |     |
| X-Achse: Anzahl der Sketches bzw. Größe der Bloomfilter                                    |     |
| Y-Achse: Relativer Fehler [%]  | 167 |
| Konfig.: Simulation im Innenstadtgebiet mit zufälligen Sensorwerten                        |     |
| 4.23 PCSA und Bloomfilter-Array:   |     |
| X-Achse: Anzahl der Sketches bzw. Größe der Bloomfilter                                    |     |
| Y-Achse: Entropie [Byte]   | 168 |
| Konfig.: Simulation im Innenstadtgebiet mit zufälligen Sensorwerten                        |     |
| 5.1 Beacon-Frame mit zusätzlicher Textnachricht  | 179 |
| 5.2 Systemarchitektur von BSMX   | 181 |
| 5.3 BSMX: Beispielanwendungen für Android  | 183 |
| 5.4 BSMX: Visualisierung der Nachbarschaftstabelle   | 184 |
| 5.5 Empfangswahrscheinlichkeiten in der Mannheimer Innenstadt                              |     |
| X-Achse: Bei der Übertragung verwendeter Funkkanal   |     |
| Y-Achse: Empfangene Beacon-Frames [%]  | 188 |
| 5.6 Konstruktion des Netzwerkgraphen   | 190 |
| 5.7 Access Points in der Mannheimer Innenstadt   | 192 |
| 5.8 Simulation der Informationsverbreitung:  |     |
| X-Achse: Simulationsrunden   |     |
| Y-Achse: Informierte Knoten [%] und abgedeckte Straßenabschnitte [%]                       | 193 |



# Tabellenverzeichnis

|  |     |
|--|-----|
| 3.1 Vergleich positionsbasierter Routingverfahren . . . . .                | 40  |
| 3.2 GRANT: Gefundene Pfade und benötigter Speicherplatz . . . . .          | 84  |
| 3.3 Face-2 und DV-Routing: Vergleich der Überbrückungsstrategien . . . . . | 105 |
| 3.4 GRANT: Übersicht der untersuchten Einflussfaktoren . . . . .           | 106 |
| 3.5 GRANT: Eigenschaften der beiden Überbrückungsverfahren . . . . .       | 107 |
| 4.1 Auszug aus der Lookup-Tabelle von PCSA-LT . . . . .                    | 131 |
| 5.1 Verwendete SSIDs in der Mannheimer Innenstadt . . . . .                | 187 |



# Abkürzungsverzeichnis

|                         |   |
|-------------------------|---|
| <b>A2DP</b> .....       | Advanced Audio Distribution Profile             |
| <b>AES</b> .....        | Advanced Encryption Standard                    |
| <b>ANT</b> .....        | Abstract Neighbour Table                        |
| <b>AODV</b> .....       | Ad hoc On-Demand Distance Vector Routing        |
| <b>TAG</b> .....        | Tiny Aggregation                                |
| <b>TiNA</b> .....       | Temporal Coherency-Aware in-Network Aggregation |
| <b>BF-Array</b> .....   | Bloomfilter-Array                               |
| <b>BF-Map</b> .....     | Bloomfilter-Map                                 |
| <b>BFP</b> .....        | Beaconless Forwarder Planarization              |
| <b>BF</b> .....         | Bloomfilter                                     |
| <b>BLR</b> .....        | Beaconless Routing                              |
| <b>BSMX</b> .....       | Beacon-Based Short Message Exchange             |
| <b>BSS</b> .....        | Basic Service Set                               |
| <b>SUMO</b> .....       | Simulation of Urban Mobility                    |
| <b>CBF</b> .....        | Contention-Based Forwarding                     |
| <b>CTS</b> .....        | Clear to send                                   |
| <b>DHCP</b> .....       | Dynamic Host Configuration Protocol             |
| <b>DSDV</b> .....       | Destination-Sequenced Distance Vector Routing   |
| <b>DSR</b> .....        | Dynamic Source Routing                          |
| <b>FM-Sketches</b> .... | Flajolet-Martin Sketches                        |
| <b>FPGA</b> .....       | Field Programmable Gate Array                   |
| <b>GFG</b> .....        | Greedy-Face-Greedy                              |

## Abkürzungsverzeichnis

---

|                |       |   |
|----------------|-------|---|
| <b>GG</b>      | ..... | Gabriel Graph                                     |
| <b>GPCR</b>    | ..... | Greedy Perimeter Coordinator Routing              |
| <b>GPSR</b>    | ..... | Greedy Perimeter Stateless Routing                |
| <b>GPS</b>     | ..... | Global Positioning System                         |
| <b>GRANT</b>   | ..... | Greedy Routing with Abstract Neighbour Table      |
| <b>GRS</b>     | ..... | Greedy Routing Scheme                             |
| <b>GSM</b>     | ..... | Global System for Mobile Communications           |
| <b>GSR</b>     | ..... | Geographic Source Routing                         |
| <b>IBSS</b>    | ..... | Independent Basic Service Set                     |
| <b>IEEE</b>    | ..... | Institute of Electrical and Electronics Engineers |
| <b>IGF</b>     | ..... | Implicit Geographic Forwarding                    |
| <b>IP</b>      | ..... | Internet Protocol                                 |
| <b>JD</b>      | ..... | Junction Detection                                |
| <b>KML</b>     | ..... | Keyhole Markup Language                           |
| <b>LAN</b>     | ..... | Local Area Network                                |
| <b>LTE</b>     | ..... | Long Term Evolution                               |
| <b>MFR</b>     | ..... | Most Forward within Radius                        |
| <b>MRD</b>     | ..... | Modified Random Direction                         |
| <b>NB-Face</b> | ..... | No-Beacon Face                                    |
| <b>NFP</b>     | ..... | Nearest Forward Progress                          |
| <b>PAN</b>     | ..... | Personal Area Networking Profile                  |
| <b>PBAP</b>    | ..... | Phone Book Access Profile                         |
| <b>PCSA</b>    | ..... | Probabilistic Counting with Stochastic Averaging  |
| <b>RFC</b>     | ..... | Request for Comments                              |
| <b>RGR</b>     | ..... | Restricted Greedy Routing                         |
| <b>RLS</b>     | ..... | Reactive Location Service                         |

|                    |  |
|--------------------|--|
| <b>RNG</b> .....   | Relative Neighbourhood Graph               |
| <b>RREP</b> .....  | Route Reply                                |
| <b>RREQ</b> .....  | Route Request                              |
| <b>RTS</b> .....   | Request to Send                            |
| <b>RWP</b> .....   | Random Waypoint                            |
| <b>SOTIS</b> ..... | Self-Organizing Traffic Information System |
| <b>SQL</b> .....   | Structured Query Language                  |
| <b>SSID</b> .....  | Service Set Identifier                     |
| <b>STBR</b> .....  | Street Topology Based Routing              |
| <b>TCP</b> .....   | Transmission Control Protocol              |
| <b>TTL</b> .....   | Time to live                               |
| <b>UDG</b> .....   | Unit Disk Graph                            |
| <b>UDP</b> .....   | User Datagram Protocol                     |
| <b>UML</b> .....   | Unified Modeling Language                  |
| <b>UMTS</b> .....  | Universal Mobile Telecommunications System |
| <b>WLAN</b> .....  | Wireless Local Area Network                |





# Kapitel 1

## Einleitung

Verschiebe nicht auf morgen, was genauso gut  
auf übermorgen verschoben werden kann.

---

– Mark Twain –

Die Möglichkeit, mit Hilfe von Computern komplexe Berechnungen durchzuführen, hat die wissenschaftliche und wirtschaftliche Entwicklung beflügelt. Die Eingangsdaten und die Ergebnisse derartiger Berechnungen werden in Speichern hinterlegt. Typische nicht flüchtige Speicher sind Festplatten und Disketten, auf denen die Daten magnetisch gespeichert werden. In der Anfangszeit der elektronischen Datenverarbeitung war der physische Austausch von Disketten und anderen Speichern die einzige Möglichkeit, die Daten von einem Computer zu einem anderen zu übertragen. Mussten beispielsweise die Daten von einer Filiale zum Hauptsitz einer Firma übertragen werden, so wurden diese auf einen transportablen Speicher kopiert und mit der Post verschickt. Dieser Vorgang wurde inzwischen dadurch vereinfacht, dass die Daten direkt per Kabel oder drahtlos zu anderen Computern übertragen werden können.

Vergleicht man den beschriebenen Datenaustausch mit der menschlichen Kommunikation, so entsprechen die Codierungs- und Verwendungsregeln, die ein Computer nutzt, um auf einen Speicher zuzugreifen, den Vorgängen im menschlichen Gehirn. Allerdings sind dem Menschen diese Vorgänge bisher nicht vollständig bekannt und auch nicht bewusst, sodass die gesprochene menschliche Sprache nicht unmittelbar auf ihnen aufbaut. Im Unterschied hierzu sind die Codierungs- und Verwendungsregeln der Speicher bekannt und können direkt an die Datenübertragung angepasst werden. Von diesen Details abgesehen, kann vereinfacht festgehalten werden, dass die Datenübertragung mit der menschlichen Sprache vergleichbar ist. Analog hierzu entspricht der Austausch von Speichern dem Austausch von Schriftstücken.

Die gesprochene Sprache wird mit Hilfe der Stimmbänder über das Medium Luft in die nähere Umgebung abgegeben und kann von anderen Menschen über das Gehör wahrgenommen werden. Zwar können Menschen auch andere Körperteile zum Kommunizieren einsetzen – beispielsweise die Hände und die Augen beim Verwenden von Gebärdensprache –, allerdings hat sich aufgrund der biologischen Voraussetzungen des menschlichen Körpers die gesprochene Sprache als die effizienteste Kommunikationsform durchgesetzt. Mit weiter entfernten Personen kann der Mensch nur anhand von Hilfsmitteln oder in Schriftform kommunizieren. Im Unterschied hierzu haben die technologischen Rahmenbedingungen dazu geführt, dass die

Datenübertragung zwischen Computern hohe Übertragungsgeschwindigkeiten erreicht, allerdings auch sehr empfindlich gegenüber Störungen ist. Aus diesem Grund erfolgt die Datenübertragung meist über ein exklusiv für diesen Zweck zur Verfügung stehendes Medium, wie beispielsweise ein abgeschirmtes Kupferkabel. Übertragen mehrere miteinander verbundene Computer gleichzeitig Signale über dieselbe Ader des Kabels, so überlagern sich diese und können ohne signifikanten Zusatzaufwand nicht mehr oder nur noch teilweise decodiert werden. Übertragungen mittels des Mediums Luft erfordern aufgrund der fehlenden Abschirmung und der größeren Störungsempfindlichkeit sogar noch mehr zusätzliche Kontrollmechanismen, wodurch die Komplexität des Vorgangs steigt und die verfügbare Bandbreite sinkt. Aus diesen Gründen erfolgt auch heute noch die überwiegende Anzahl der Übertragungen zwischen Computern drahtgebunden.

Allerdings ist die Bedeutung der drahtlosen Kommunikation aufgrund der stark wachsenden Anzahl von mobilen Computern wie Smartphones und Laptops erheblich gestiegen. Dennoch erfolgt auch weiterhin der Großteil der Vernetzung drahtgebunden, sodass die mobilen Computer üblicherweise nur über einen Zugriffspunkt kabellos mit der bestehenden Infrastruktur verbunden werden. Ein Smartphone verbindet sich beispielsweise mit der nächstliegenden Basisstation des jeweiligen Mobilfunkbetreibers und kann dann über die Infrastruktur des eigenen Anbieters sowohl die Netzwerke der anderen Anbieter als auch das Internet erreichen. In den meisten Smartphones ist zusätzlich auch ein WLAN-Modul verbaut, welches üblicherweise für einen vom Mobilfunkanbieter unabhängigen Internetzugang verwendet wird. Dieses Beispiel zeigt, dass die meisten drahtlosen Übertragungstechnologien im Endbenutzerbereich fast ausschließlich dazu verwendet werden, mobile Geräte mit der bereits bestehenden Infrastruktur zu verbinden.

Aus Sicht der Nutzer ist diese Lösung zwar zufriedenstellend, da alle bisherigen Anwendungen problemlos weiter verwendet werden können, gleichzeitig wird aber durch diese Entwicklung nicht das volle Potenzial der drahtlosen Technologien ausgeschöpft. So sind etwa die üblicherweise in einem Smartphone verbauten Funktechnologien rein technisch gesehen in der Lage, auch ohne Infrastruktur Daten direkt mit anderen Geräten in der näheren Umgebung auszutauschen. Allerdings kann diese Funktion aufgrund fehlender Treiber, Netzwerkprotokolle und Anwendungen von den Benutzern nur eingeschränkt oder gar nicht verwendet werden. Ein weiteres großes Problem ist in diesem Zusammenhang, dass insbesondere die Hersteller von Smartphones keine tief greifende Veränderung des Systems vorsehen und daher auch die Installation von zusätzlichen Netzwerkprotokollen oder alternativen Treibern nur über Umwege möglich ist.

In der wissenschaftlichen Literatur der letzten Jahre wurde immer wieder die Vision eines dezentral organisierten drahtlosen Internets aufgegriffen, welches im Optimalfall völlig unabhängig von der bestehenden Infrastruktur funktioniert. Eine wesentliche Funktion eines solchen drahtlosen Internets besteht darin, dass jeder Knoten des Netzwerks mit jedem anderen Knoten Daten austauschen kann. Falls Daten zwischen zwei Knoten ausgetauscht werden sollen, die sich nicht innerhalb ihrer jeweiligen Funkreichweite befinden, muss auf andere

---

erreichbare Knoten zurückgegriffen werden, welche die Daten dann Schritt für Schritt zum Zielknoten weiterleiten. Die Funktionalität, die hierfür benötigten Pfade zwischen den Kommunikationsteilnehmern zu finden und die Daten entlang dieser Pfade zu übertragen, ist im Routingprotokoll zusammengefasst. Alle Netzwerkschichten und Dienste sollten an die Herausforderungen eines drahtlosen und mobilen Internets angepasst werden. Prinzipiell kann ein mit WLAN ausgestatteter Computer, sofern der Hersteller geeignete Treiber für das verwendete Betriebssystem anbietet, bereits seit Jahren ein Ad-hoc-Netzwerk aufbauen und bei entsprechender Konfiguration auch Daten mit anderen Geräten innerhalb der Funkreichweite austauschen. Auch wenn noch zahlreiche andere Herausforderungen gemeistert werden müssen, kann das Routingprotokoll, welches den Datenaustausch auch mit nicht benachbarten Geräten ermöglicht, doch als Kernstück eines drahtlosen Internets bezeichnet werden.

In bestimmten Einsatzszenarien kann nicht davon ausgegangen werden, dass durchgehend eine zuverlässige Verbindung zur Infrastruktur möglich ist. Beispielsweise werden mit Sensoren ausgestattete Kleinstcomputer für die Tierbeobachtung in freier Wildbahn eingesetzt. Die hierbei erfassten Daten werden zwischen den Computern direkt über eine Funktechnologie ausgetauscht und weiterverarbeitet. Für andere Anwendungen ist wiederum der Umweg über die Infrastruktur zu zeitintensiv. Soll beispielsweise ein Auto über einen möglicherweise bevorstehenden Zusammenstoß mit einem anderen Fahrzeug informiert werden, so muss diese Warnung so schnell wie möglich direkt übermittelt werden. Für derartige Szenarien existieren zahlreiche Ansätze, die ohne Infrastruktur auskommen und rein dezentral organisiert sind. Allerdings sind diese Strategien überwiegend in monolithischen Systemen implementiert und speziell für den jeweiligen Anwendungsfall optimiert. Soll beispielsweise die Aktivität eines Vulkans beobachtet werden, so werden mehrere Kleinstcomputer mit den benötigten Sensoren ausgestattet und an relevanten Positionen platziert. Diese Computer bauen dann ein Ad-hoc-Netzwerk auf und tauschen hierüber die erfassten Sensordaten aus. Da es sich um ein geschlossenes System handelt, kann jederzeit kontrolliert werden, welche Geräte welche Daten übermitteln. In diesem Fall ist es nicht notwendig, das System in Netzwerkschichten zu unterteilen und offene Schnittstellen zu implementieren. Auch viele andere Aspekte, wie beispielsweise eine Überlastkontrolle oder die Überwachung der Netztopologie, sind in solchen geschlossenen Systemen meist nicht erforderlich. Daher können diese Ansätze zwar als Grundlage für Weiterentwicklungen verwendet, aber nicht ohne jede Anpassung direkt auf offene Systeme übertragen werden. Da in den beschriebenen Fällen alle Computer des Netzwerks gemeinsam eine Aufgabe wie beispielsweise die Tierüberwachung meistern, werden die entsprechenden Verfahren im Folgenden als kooperative Anwendungen bezeichnet.

Die vorliegende Arbeit wird von der Vision getragen, dass alle mit einer Funktechnologie ausgestatteten Geräte diese nicht ausschließlich für den direkten Zugriff auf die Infrastruktur verwenden, sondern zusätzlich universell einsetzbare Ad-hoc-Netzwerke aufbauen. Dadurch soll nicht nur eine drahtlose Erweiterung des Internets ermöglicht werden, sondern auch das Erstellen von flexibel einsetzbaren kooperativen Anwendungen mit offenen Schnittstellen auf Endkundengeräten wie beispielsweise Smartphones.

Im Folgenden wird die Struktur der Dissertation vorgestellt. Die Grundlagen der drahtlosen Kommunikation in Ad-hoc-Netzwerken werden in Kapitel 2 vermittelt. Hierbei werden sowohl ausgewählte Routingverfahren als auch exemplarische kooperative Anwendungen vorgestellt.

Zu Beginn von Kapitel 3 wird der Frage nachgegangen, wie die Leistungsfähigkeit eines Ad-hoc-Routingprotokolls ermittelt werden kann. Üblicherweise werden hierfür Netzwerksimulatoren eingesetzt, die möglichst akkurat alle Netzwerkschichten simulieren sollen. Allerdings ist auch schon für ein kleines Simulationsgebiet ein immenser Berechnungsaufwand notwendig. Um dennoch untersuchen zu können, ob ein Routingverfahren für alle Kommunikationspartner eines großen Netzwerks zuverlässig einen Pfad finden kann, wird ein speziell für diesen Zweck entwickelter Simulator vorgestellt. Neben der effizienten Berechnung der Pfadsuche für alle Netzteilnehmer kann dieser Simulator auch die anhand von Satellitenbildern ermittelten Funkhindernisse berücksichtigen. Des Weiteren wird in diesem Kapitel mit *Greedy Routing with Abstract Neighbour Table* ein neues positionsbasiertes Ad-hoc-Routingprotokoll präsentiert, dessen Leistungsfähigkeit mit Hilfe einer ausführlichen Simulationsstudie ermittelt und mit anderen bestehenden Ansätzen verglichen wird.

Oft bauen kooperative Anwendungen auf Netzwerken ohne klassische Routingprotokolle auf. In solchen Fällen kann jeder Knoten nur mit seinen direkten Nachbarn in Funkreichweite kommunizieren. Eine häufige Aufgabenstellung ist die dezentrale Datenverarbeitung. Sollen beispielsweise die Sensordaten aller Knoten miteinander kombiniert werden, so besteht ein simpler Lösungsansatz darin, dass jeder Knoten schrittweise die ihm bekannten Sensordaten an seine Nachbarn überträgt. Durch jede Iteration dieser Strategie wird der Radius der bekannten Sensorwerte größer, bis schließlich jeder Knoten alle Sensorwerte kennt. In Netzwerken mit einer hohen Knotendichte kann allerdings die Datenmenge bereits nach einer Iteration so groß sein, dass sie nicht mehr in einem Paket übertragen werden kann. Diese Problematik wird durch den Umstand verstärkt, dass eben nicht nur die Sensorwerte übertragen werden müssen, sondern auch noch die Information, welcher Sensorwert zu welchem Knoten gehört. Daher wird in einigen – später noch im Detail besprochenen – Publikationen vorgeschlagen, anstelle der eigentlichen Daten eine probabilistische Datenstruktur auszutauschen, die ausschließlich statistische Merkmale der Daten und des zugehörigen Knotens speichert. Diese Informationen benötigen vergleichsweise wenig Speicherplatz und können von den Empfängern beispielsweise verwendet werden, um näherungsweise die Summe oder den Durchschnitt aller Werte zu bestimmen. Auch wenn bei diesem Ansatz kein exaktes Ergebnis ermittelt werden kann, so ist die erreichbare Genauigkeit für viele Anwendungen doch ausreichend. In Kapitel 4 werden der eben skizzierte Algorithmus und die hierfür verwendete Datenstruktur im Detail beschrieben. Um Aussagen darüber treffen zu können, welche Genauigkeit bei welcher Konfiguration erreicht werden kann, wird in diesem Kapitel die Datenstruktur mit Hilfe eines Simulators empirisch evaluiert. Hierbei wird sowohl die Genauigkeit der Datenstruktur an sich bestimmt als auch die Genauigkeit, die sich beim Einsatz der Datenstruktur für die dezentrale Aggregation ergibt. Darüber hinaus werden mit *Bloomfilter-Maps* und *Bloomfilter-Arrays* zwei neue,

---

alternative Datenstrukturen vorgeschlagen, deren Genauigkeiten ebenfalls empirisch ermittelt werden.

Die zu Beginn des Kapitels beschriebene Entwicklung der elektronischen Datenübertragung hat dazu geführt, dass auch heute noch der Großteil des Datenverkehrs drahtgebunden übertragen wird und Funktechnologien meist nur für den bequemen Zugriff auf die bestehende Infrastruktur verwendet werden. Aufgrund der Störungsempfindlichkeit und der erhöhten Komplexität von drahtlosen Übertragungen wird sich diese Vorgehensweise auch in den nächsten Jahren voraussichtlich nicht ändern. Hierdurch wird aber die in klassischen Netzwerken propagierte Ende-zu-Ende-Kommunikation auch auf die drahtlosen Netzwerke ausgeweitet. Wie im Zusammenhang mit den probabilistischen Datenstrukturen bereits beschrieben, gibt es einige kooperative Anwendungen, die eine rein dezentrale Strategie verfolgen. In diesen Fällen kommunizieren Knoten, weil sie gemeinsam eine Aufgabe lösen, und nicht, weil sie explizit adressiert werden. Auch in der menschlichen Kommunikation sind gemeinsame Interessen häufig der eigentliche Grund, aus dem eine Konversation geführt wird. Bleiben beispielsweise mehrere Personen im Fahrstuhl stecken, so sind für jeden Gast automatisch die anderen Personen in unmittelbarer Nähe von besonderem Interesse, unabhängig davon, ob diese persönlich bekannt sind oder nicht. Eine Grundannahme dieser Arbeit besteht deshalb darin, die Funkreichweite als eine Art Analogie zu der menschlichen Sprechdistanz zu betrachten und daher alle anderen Geräte in unmittelbarer Nähe als potenziell interessante Kommunikationspartner aufzufassen. Daher wird in Kapitel 5 der Frage nachgegangen, wie handelsübliche Geräte, die mit einer Funktechnologie ausgestattet sind, so modifiziert werden können, dass ein direkter Datenaustausch ohne Infrastruktur möglich wird. Dabei sollte allerdings die ursprüngliche Funktionalität der Geräte nicht über Gebühr beeinträchtigt werden. In diesem Kapitel wird mit *Beacon-Based Short Message Exchange* eine Strategie vorgestellt, die den Austausch von Kurznachrichten zwischen uneinheitlich konfigurierten WLAN-Geräten ermöglicht. Zwar können dabei nur vergleichsweise kleine Datenmengen übertragen werden, dafür können aber die Geräte, wie beispielsweise Access Points oder Smartphones, nahezu ohne jede Einschränkung wie bisher verwendet werden. Neben der grundsätzlichen Funktionsweise wird in diesem Kapitel auch die Implementierung eines Prototyps beschrieben. In dicht bewohnten Innenstadtgebieten gibt es eine hohe Anzahl von Access Points, deren Übertragungen auch in benachbarten Wohnungen und auf der Straße empfangen werden können. Diese Geräte könnten prinzipiell mit BSMX ausgestattet werden und so eine Art lose zusammenhängendes Ad-hoc-Netzwerk bilden. Um die zu erwartende Konnektivität eines solchen Netzwerks abschätzen zu können, wurden umfangreiche Messungen in der Mannheimer Innenstadt durchgeführt. Der genaue Versuchsaufbau und die darauf basierenden Simulationsergebnisse werden ebenfalls in Kapitel 5 vorgestellt.

In Kapitel 6 werden schließlich die wichtigsten Ergebnisse dieser Arbeit zusammengefasst. Des Weiteren wird ein kurzer Ausblick über mögliche Weiterentwicklungen gegeben.



## Kapitel 2

# Grundlagen der drahtlosen Kommunikation in Ad-hoc-Netzwerken

Holzhacken ist deshalb so beliebt, weil man bei dieser Tätigkeit den Erfolg sofort sieht.

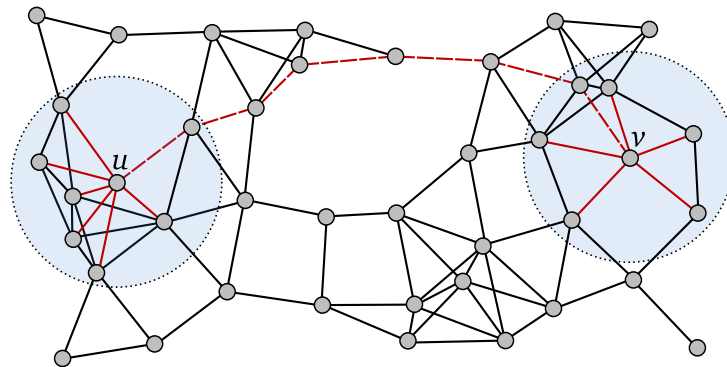
---

– Albert Einstein –

Ist ein Gerät mit einer drahtlosen Übertragungstechnik ausgestattet, so kann es passend ausgestrahlte elektromagnetische Wellen eines spezifizierten Wellenspektrums bei ausreichender Empfangsleistung wieder in Nutzdaten umwandeln. Welche Empfangsleistung hierfür ausreicht, hängt von der Codierungsmethode, der verwendeten Übertragungstechnik und der genutzten Hardware ab. Sieht man von Fehlerursachen wie beispielsweise Wellenüberlagerungen durch Störquellen ab, so ergibt sich die maximale Funkreichweite einer Übertragungstechnik aus der Sendeleistung und Abstrahlcharakteristik des Senders sowie aus der minimal notwendigen Empfangsleistung des Empfängers. Im Normalfall werden in einem Ad-hoc-Netzwerk Rundstrahlantennen (zum Beispiel Stabantennen) eingesetzt, deren Abstrahlcharakteristik in der Horizontalen in alle Richtungen nahezu gleich ist. Somit ergibt sich in einer störungsfreien Umgebung ein annähernd kreisförmiges Gebiet um den Sender, in dem andere Geräte die Daten des Senders erfolgreich empfangen können. Der genaue Einfluss von Störungsquellen ist vom Einsatzort abhängig und kann nicht allgemeingültig modelliert werden. Des Weiteren sind die meisten Beeinträchtigungen zeitlich nicht konstant, sodass in einem realen Ad-hoc-Netzwerk ein Empfänger typischerweise nur einen bestimmten Prozentsatz aller gesendeten Datenpakete eines Senders erfolgreich empfangen kann [142]. Aus diesen Gründen ist es nicht möglich, anhand der Gerätepositionen einen statischen und universellen Netzwerkgraphen zu generieren, der alle denkbaren Störungen und Beeinflussungen korrekt widerspiegelt. Zur Modellierung von Ad-hoc-Netzwerken wird deshalb meist auf die idealisierte kreisförmige Signalausbreitung zurückgegriffen, wobei der Radius dieses Kreises im Folgenden als Funkreichweite bezeichnet wird.

Ein Graph ist definiert als das Tupel  $G = (V, E)$ , wobei  $V$  die Menge der Knoten und  $E$  die Menge der Kanten ist. In einem Netzwerkgraphen entsprechen die Knoten den am Netzwerk teilnehmenden Geräten und die Kanten den Kommunikationsverbindungen zwischen den teilnehmenden Geräten. In einem Funknetzwerk ist eine Kommunikationsverbindung vom Knoten  $v_1 \in V$  zu dem Knoten  $v_2 \in V$  immer dann gegeben, wenn der euklidische Abstand  $d(v_1, v_2)$

kleiner oder gleich der Funkreichweite von  $v_1$  ist. Dementsprechend lässt sich die Menge der Kanten  $E$  erzeugen, indem für jeden Knoten  $v_1 \in V$  jeweils eine ausgehende Kante zu allen anderen Knoten  $v_2 \in V$  mit  $d(v_1, v_2)$  kleiner oder gleich der Funkreichweite von  $v_1$  erstellt wird. Da der euklidische Abstand kommutativ ist, kann der hierbei erzeugte gerichtete Graph für Netzwerke mit einheitlicher Funkreichweite verlustfrei in einen ungerichteten Graphen überführt werden. Derartige Graphen, in denen alle Knoten eine Kante zu allen anderen Knoten innerhalb eines einheitlichen Radius besitzen, werden *Unit Disk Graphs* (UDGs) [37] genannt. Sie bilden die Grundlage zahlreicher Algorithmen aus dem Bereich der Ad-hoc-Kommunikation [75]. Abbildung 2.1 zeigt anhand eines UDG die Kommunikationsmöglichkeiten der Netzteilnehmer. Für die Knoten  $u$  und  $v$  sind exemplarisch die direkt erreichbaren Nachbarknoten mit einer roten Kante markiert.



**Abbildung 2.1:** Ein die Kommunikationsmöglichkeiten der Netzteilnehmer widerspiegelnder Unit Disk Graph.

Jeder Knoten eines solchen Netzwerkgraphen kann mit allen über eine Kante verbundenen Knoten direkt kommunizieren. Damit auch nicht benachbarte Knoten miteinander Daten austauschen können, muss in dem Graphen ein Pfad gefunden werden, der beide Kommunikationspartner miteinander verbindet. In Abbildung 2.1 sind die Kanten eines möglichen Pfades zwischen Knoten  $u$  und  $v$  als rot gestrichelte Linien hervorgehoben. Die Knoten auf diesem Pfad sind an der Ende-zu-Ende-Kommunikation unbeteiligt und fungieren ausschließlich als Vermittler. Daher wird diese Art des Datenaustauschs Ende-zu-Ende-Kommunikation genannt. Die Vorgänge, einen Pfad zwischen Kommunikationspartnern zu suchen und die Daten entlang dieses Pfades weiterzuleiten, werden unter der Bezeichnung *Routing* zusammengefasst. Üblicherweise soll in einem solchen Szenario der Inhalt der weitergeleiteten Pakete von den Vermittlern weder interpretiert noch modifiziert werden, sodass die sogenannten Nutzdaten auf den nicht adressierten Vermittlern gar nicht erst an die lokalen Anwendungen weitergeleitet werden. Alternativ zu dieser Ende-zu-Ende-Kommunikation, bei der explizit der Empfänger der Daten adressiert wird, besteht auch die Möglichkeit, auf eine Adressierung zu verzichten und den Empfängern innerhalb der Funkreichweite die Entscheidung über das weitere Vorgehen zu überlassen. Jeder Empfänger kann in diesem alternativen Szenario die Daten des



gesendeten Pakets interpretieren; kennt also den Aufbau und die Codierung der übertragenen Nutzdaten. Der Empfänger kann dann gemäß seiner lokalen Sichtweise frei entscheiden, ob er die Daten vor dem Weiterleiten modifiziert, indem er beispielsweise die Daten mit anderen lokal vorliegenden Informationen kombiniert oder sogar auf eine weitere Verbreitung verzichtet. In diesem Fall sind alle Knoten gleichberechtigt und versuchen gemeinsam eine Aufgabenstellung zu lösen. Daher werden derartige Strategien im Folgenden als kooperative Anwendungen bezeichnet. Beide Vorgehensweisen werden in den folgenden zwei Abschnitten separat voneinander näher besprochen.

## 2.1 Ausgewählte Routingverfahren

Die Herausforderung beim Routing liegt darin, dass jeder einzelne Knoten nur sich selbst kennt und keine Kenntnis des gesamten Netzwerkgraphen besitzt. Dementsprechend kann jeder Knoten den Netzwerkgraphen oder Teile dessen nur durch die Kooperation mit anderen Netzteilnehmern bestimmen. Der erste Schritt in diesem Prozess besteht immer darin, dass jeder Knoten die benachbarten Knoten innerhalb seiner Funkreichweite über seine Existenz informiert. Im Regelfall hat jeder Knoten einen eindeutigen Identifikator und kann mit dessen Übertragung seine Nachbarn über seine Existenz informieren. Hierbei ist der Umstand zu beachten, dass in Ad-hoc-Netzwerken mit Rundstrahlantennen jede gesendete Nachricht immer von allen anderen Stationen innerhalb der Funkreichweite empfangen werden kann. Erst auf der Empfängerseite kann festgestellt werden, für welche Station die Nachricht bestimmt ist. Somit kann ein Knoten eines Ad-hoc-Netzwerks alle direkten Nachbarn mit nur einer Nachricht informieren. Nach diesem Schritt wissen alle Knoten, welche anderen Knoten sich in ihrer Funkreichweite befinden, und können dieses Wissen wiederum mit anderen Knoten austauschen. Mit jeder Wiederholung dieses Vorgangs wird der bekannte Teil des Graphen vergrößert, bis schließlich jeder Knoten die Topologie des gesamten Netzwerks kennt. Dieses Wissen kann dann jeder einzelne Knoten nutzen, um zu jedem anderen Knoten den optimalen Pfad zu bestimmen.

Bei sehr kleinen Netzwerken ist die Bestimmung des gesamten Netzwerkgraphen ein praktikabler Ansatz, für größere Netzwerke ist jedoch die Anzahl der Nachrichten, die dabei ausgetauscht werden müssten, viel zu hoch. Des Weiteren sind drahtlos kommunizierende Geräte in der Regel mobil, wodurch der ermittelte Graph sehr schnell veraltet und ständig aktualisiert werden müsste. Es ist deshalb die Kernaufgabe eines Routingverfahrens, mit möglichst wenig Datenaustausch Pfade zwischen Kommunikationspartnern zu finden und diese selbst in mobilen Netzwerken aufrechtzuerhalten beziehungsweise automatisch zu erneuern.

Bei einer nahezu statischen Topologie werden meist einmalig Daten ausgetauscht, die es dann jedem Knoten ermöglichen, mit jedem anderen Knoten zu kommunizieren. Nach diesem Vorgang kennt zwar jeder Knoten nicht das gesamte Netzwerk, er weiß aber, an welchen Nachbarn er die Daten weiterleiten muss, um einen bestimmten Zielknoten zu erreichen. Die hierfür verwendete Datenstruktur wird *Routingtabelle* genannt. Sollte es zu einer geringfügigen Änderung des Netzwerks kommen, so müssen nur die betroffenen Knoten ihre Routingtabel-

len aktualisieren. Als *proaktiv* werde Routingverfahren bezeichnet, die kontinuierlich einen Pfad von jedem Knoten zu jedem anderen Knoten des Netzwerks bereitstellen, auch wenn der einzelne Knoten jeweils nur den nächsten Weiterleiter kennt. Der kontinuierlich benötigte Kommunikationsaufwand, um die Routingtabellen immer aktuell zu halten, ist in hoch mobilen Umgebungen enorm hoch und kann bei großen Netzwerken schnell die verfügbare Gesamtbandbreite übersteigen. Deshalb verfolgen neuere Verfahren einen *reaktiven* Ansatz, indem sie erst im Bedarfsfall einen Pfad zwischen zwei Knoten bestimmen. Der Betrieb dieser Verfahren verursacht einen signifikant geringeren kontinuierlichen Datenaustausch als der von proaktiven Verfahren. Allerdings wird der Beginn der eigentlichen Kommunikation durch die Pfadsuche geringfügig verzögert. Alle im Folgenden vorgestellten Verfahren zur Wegewahl sind reaktive Ansätze.

Die meisten Verfahren zur Wegewahl, insbesondere in drahtgebundenen Netzwerken, versuchen Pfade zwischen Knoten zu bestimmen, indem sie die Topologie des Netzwerks durch Nachrichtenaustausch ganz oder teilweise ermitteln. Verfügen allerdings alle Geräte des Netzwerks über ein Positionierungssystem, so können auch die Positionen der Knoten bei der Pfadsuche berücksichtigt werden. Derartige Ansätze werden deshalb positionsbasierte Routingverfahren genannt. Die gängigen positionsbasierten Verfahren besitzen vier gemeinsame Voraussetzungen:

1. Alle Knoten verwenden Positionsinformationen mit einem netzwerkweit einheitlichen Koordinatensystem.
2. Jeder Knoten kennt seine eigene Position.
3. Jeder Knoten kennt die Positionen seiner direkten Nachbarn in Funkreichweite.
4. Die Position des Zielknotens, zu dem ein Pfad gefunden werden soll, ist bekannt.

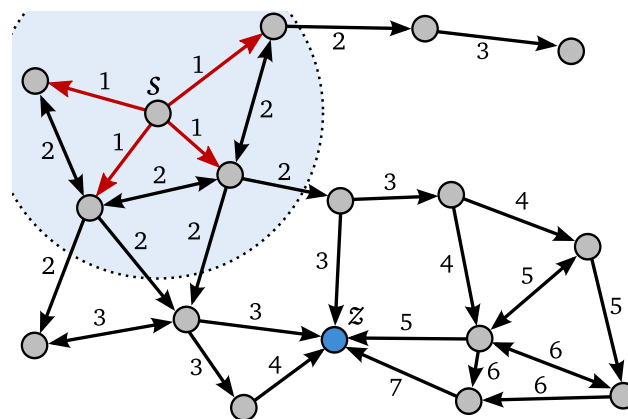
Es existieren Verfahren, die es einem Knoten erlauben, seine eigene Position relativ zu den anderen Knoten in seiner Nachbarschaft zu bestimmen. Die erste Voraussetzung stellt entsprechend sicher, dass alle Knoten ein gemeinsames Koordinatensystem mit absoluten Positionen verwenden. Indem jeder Knoten mit einem Positionierungssystem ausgestattet wird, kann die zweite Voraussetzung erfüllt werden. Hierfür bietet sich das verbreitete *Global Positioning System* (GPS) [85] an, welches auch von allen gängigen Navigationssystemen verwendet wird. Sollte GPS nicht verfügbar sein oder sich aufgrund des Energieverbrauchs nicht eignen, so ist es möglich, mit Hilfe von Laufzeitmessungen einer drahtlosen Übertragungstechnik die Abstände zwischen den Netzteilnehmern abzuschätzen. Diese Abstände können dann in ein für alle Knoten einheitliches Koordinatensystem überführt werden [32]. Eine Übersicht über derartige Verfahren, die häufig für den Einsatz in Sensornetzen konzipiert sind, ist in [28] zu finden. Innerhalb von Gebäuden sind Laufzeitmessungen aufgrund der vielen Hindernisse und Reflexionen weniger gut geeignet. Ebenfalls können in Gebäuden meist nicht ausreichend viele GPS-Satelliten empfangen werden. Es besteht allerdings die Möglichkeit, in diesen Szenarien die Position anhand der bestehenden WLAN- oder Bluetooth-Infrastruktur zu bestimmen

[90]. Die dritte Voraussetzung kann erfüllt werden, indem jeder Knoten die eigene Position an alle anderen Knoten in Funkreichweite übermittelt. Die letzte Voraussetzung kann je nach Einsatzszenario unterschiedlich erfüllt werden. Im günstigsten Fall ist die Zielposition a priori bekannt und muss nicht ermittelt werden. In einem Fahrzeug-zu-Fahrzeug-Netzwerk könnte beispielsweise jedes Fahrzeug über eine Liste von fest positionierten Access Points verfügen, mit denen kommuniziert werden kann. Für den universellen Einsatz ist allerdings ein Mechanismus notwendig, der die Position eines Knotens anhand dessen Identifikators ermitteln kann. In Ad-hoc-Netzwerken ist davon auszugehen, dass hierfür kein zentraler Service existiert und folglich die Position des Zielknotens nur durch eine netzwerkweite Suche ermittelt werden kann. Allerdings können die Positionen, je nach Mobilität des Netzwerks, zwischengespeichert und hierarchisch strukturiert werden, sodass nicht immer eine netzweite Suche erforderlich ist. Eine Übersicht existierender Verfahren findet sich beispielsweise in [65] und [41].

In den folgenden Unterabschnitten werden sowohl topologiebasierte als auch positionsbasierte Routingverfahren präsentiert und im Detail diskutiert.

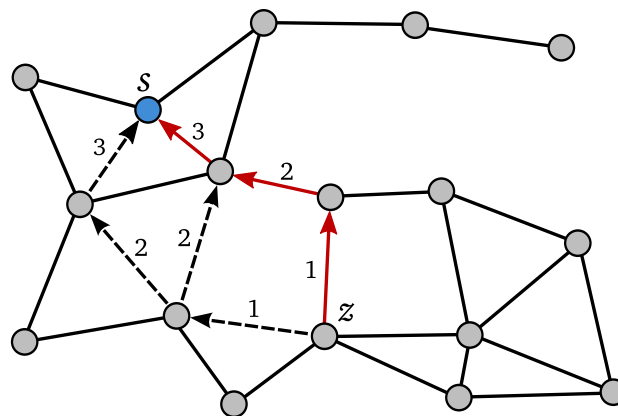
### 2.1.1 Ad hoc On-Demand Distance Vector Routing

*Ad hoc On-Demand Distance Vector Routing* (AODV) [122] ist ein topologiebasiertes Routingverfahren für Ad-hoc-Netzwerke und sucht reaktiv nach benötigten Pfaden. Da AODV sehr gut untersucht und weitverbreitet ist, wird es hier stellvertretend für die Klasse der topologiebasierten Routingverfahren vorgestellt. Wie sein proaktiver Vorgänger *Destination-Sequenced Distance Vector Routing* (DSDV) [121] basiert AODV auf einer verteilt berechneten Variante des *Bellman-Ford-Algorithmus* [21, 62], welcher ausgehend von einem Startknoten die kürzesten Pfade zu allen anderen Knoten in einem kantengewichteten Graphen bestimmen kann. Typischerweise wird die Suche aber abgebrochen, sobald ein Pfad zum Zielknoten gefunden wurde.



**Abbildung 2.2:** Die Suche eines Pfads von Knoten  $s$  zu Knoten  $z$  mit Hilfe des AODV-Protokolls.

Wird auf einem Knoten  $s$  ein bisher unbekannter Pfad zu Knoten  $z$  benötigt, so startet  $s$  eine Suche, indem er an alle benachbarten Knoten einen sogenannten *Route Request* (RREQ) sendet. Dieser Vorgang wird in Abbildung 2.2 dargestellt. Das hierbei übertragene RREQ-Paket enthält unter anderem die Adressen des Start- und Zielknotens, die sogenannte Broadcast-ID und die Länge des bisher zurückgelegten Pfads (*PathLen*). Als Adresse kann eine IP-Adresse oder ein anderer Identifikator verwendet werden. Im Folgenden wird deshalb von einer Knoten-ID gesprochen, die für einen Knoten  $v$  mit  $v_{ID}$  abgekürzt wird. Jeder Knoten  $v$  verwaltet seine eigene Broadcast-ID  $v_{BID}$ , die nach dem Starten einer Pfadsuche um eins erhöht wird. Hierdurch ist jede Pfadsuche eines Knotens  $v$  durch das Tupel  $(v_{ID}, v_{BID})$ , das im entsprechenden RREQ mitgeführt wird, eindeutig identifiziert. Jeder Empfänger eines RREQ-Pakets kennt nun den Knoten, der die Suche initiiert hat, und weiß, dass der Knoten, von dem er den RREQ empfangen hat, auf dem kürzesten Pfad zum Startknoten liegt. Durch das entsprechende Feld im RREQ-Paket kennt er auch die Länge des Pfads zum Startknoten. Hierdurch kann er einen entsprechenden Eintrag in seine Routingtabelle einfügen beziehungsweise einen existierenden Eintrag aktualisieren. Des Weiteren überprüft der Empfänger, ob er selbst der gesuchte Knoten ist oder einen Pfad zu diesem kennt. Sollte keine der beiden Bedingungen erfüllt sein, so erhöht der Knoten die *PathLen* des RREQ-Pakets und schickt es ansonsten unverändert an all seine direkten Nachbarn. Um hierbei Schleifen zu vermeiden, werden schon bekannte RREQ-Pakete bei einem erneuten Empfang ignoriert. In Abbildung 2.2 wird die Übertragungsrichtung der RREQ-Pakete und deren zeitlicher Ablauf dargestellt. Da die Suche nach dem Zielknoten verteilt und parallel abläuft, endet sie erst, wenn jeder Knoten innerhalb des im RREQ-Paket hinterlegten Suchradius erreicht wurde. Sollte der gesuchte Knoten nicht im ersten Durchlauf gefunden worden sein, wird die Suche mit einem schrittweise erhöhten Suchradius so lange wiederholt, bis der Knoten gefunden oder das gesamte Netzwerk durchsucht wurde.



**Abbildung 2.3:** Der Rückweg von Knoten  $z$  zu Knoten  $s$  anhand des zuvor ermittelten Hinwegs.

Erreicht schließlich eine Kopie des RREQ-Pakets den Zielknoten beziehungsweise einen Knoten, der einen Pfad zu diesem kennt, so ist durch die auf dem Hinweg erstellten Routingtabel-

leneinträge der kürzeste Rückweg zum Startknoten bekannt. Jeder Knoten kennt dabei immer nur den nächsten Weiterleiter und die verbleibende Länge des Pfads zum Startknoten. Allerdings wurden in den Routingtabellen bisher noch keine Einträge für den Pfad vom Startknoten zum Zielknoten hinterlegt. Deshalb wird ein sogenanntes *Route Reply* (RREP) entlang des Hinwegs zurückgeschickt. In dem betrachteten Beispiel gibt es mehrere Pfade der gleichen Länge, die zum Zielknoten führen. Über welchen Pfad das erste RREP-Paket in einem solchen Szenario eintrifft, hängt von äußeren Faktoren wie beispielsweise der Netzwerkauslastung ab. In Abbildung 2.3 wurden die drei möglichen Rückwege hervorgehoben, wobei die Kantenbeschriftung wieder den zeitlichen Ablauf dokumentiert. Im Gegensatz zur Suche des Hinwegs, bei der die RREQ-Pakete immer an alle Nachbarn geschickt werden (*Broadcast*), werden die RREP-Pakete auf dem Rückweg immer nur gezielt an die in den Routingtabellen jeweils hinterlegten Nachbarn geschickt (*Unicast*). Analog zum Hinweg wird während des Rückwegs auf den passiert Knoten der Pfad zum Zielknoten in den Routingtabellen hinterlegt.

Neben der Pfadsuche beinhaltet AODV zahlreiche weitere Funktionen. Beispielsweise werden Schleifen in der Paketweiterleitung durch den Einsatz von Sequenznummern verhindert. Da AODV explizit für mobile Ad-hoc-Netzwerke konzipiert wurde, beinhaltet es auch Mechanismen zur Korrektur ungültig gewordener Pfade. Jeder Knoten kann durch den Nachrichtenaustausch mit seinen Nachbarn in Form der regelmäßig übertragenen Statuspakete oder der weitergeleiteten Datenpakete überprüfen, ob die Einträge in seiner Routingtabelle weiterhin den lokalen Gegebenheiten entsprechen. Sollte dies nicht der Fall sein, kann mit Hilfe eines lokal arbeitenden Reparaturmechanismus der Pfad aktualisiert werden. Falls der Reparaturversuch erfolglos bleibt, werden die noch erreichbaren Knoten des Pfades informiert.

Der grundsätzliche Ablauf der Pfadsuche geht auf [40] zurück und wird in modifizierter Form auch in *Dynamic Source Routing* (DSR) [83] verwendet. Im Unterschied zu AODV werden bei DSR die entdeckten Pfade als Knotenliste an die beiden Kommunikationspartner übertragen. Bei der eigentlichen Datenübertragung wird dann der Pfad zum jeweiligen Zielknoten explizit im Paketkopf mitgeschickt. Im Vergleich zu den dezentral organisierten Routingtabellen von AODV führt die zusätzliche Übertragung des Pfads in jedem Paket mit Nutzdaten zu einem erhöhten Bandbreitenverbrauch. Ein weiterer Vorteil von AODV liegt darin, dass kleinere topologische Veränderungen durch eine lokale Anpassung der betroffenen Routingtabellen kompensiert werden können, ohne hierbei die Pfadanpassungen bis zu den jeweiligen Kommunikationspartnern propagieren zu müssen.

Schon kurz nach der Veröffentlichung von AODV wurde das Protokoll für den Linux-Kernel 2.2 implementiert [129, 1]. Um die Interoperabilität zwischen unterschiedlichen Implementierungen zu gewährleisten, wurde AODV in einem *Request for Comments* (RFC) detailliert dokumentiert [120]. Inzwischen gibt es zahlreiche weitere Linux-Implementierungen [117, 34, 33]. Unter Microsoft Windows kann die in Java realisierte Implementierung der Universität Bremen verwendet werden [96]. Des Weiteren existieren auch Implementierungen für Sensorknoten [18] und FPGAs [126]. Für Simulationszwecke ist AODV auch in nahezu allen Netzwerksimulatoren enthalten.

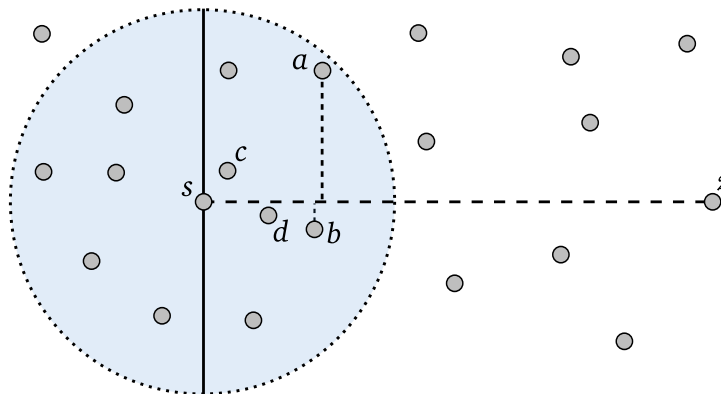
### 2.1.2 Positionsbasierte Greedy-Algorithmen

Greedy-Algorithmen [39] bilden in der Informatik eine spezielle Klasse von Algorithmen. Sie haben gemeinsam, dass sie die Aufgabenstellung schrittweise abarbeiten und hierbei unter den verfügbaren nächsten Schritten denjenigen auswählen, der zum Zeitpunkt der Wahl den größten Gewinn beziehungsweise Fortschritt verspricht. Um die Eignung der möglichen Folgeschritte einschätzen zu können, wird meist eine auf die Aufgabenstellung angepasste Bewertungsfunktion verwendet. Da der nächste Schritt ausschließlich anhand seines Fortschritts bezüglich der Aufgabenstellung ausgewählt wird – also ohne Kenntnis der Ergebnisse aller möglichen Folgeschritte –, können Greedy-Algorithmen nicht jede Problemstellung optimal lösen. Doch selbst für Aufgabenstellungen, die sich nicht exakt und optimal mit Algorithmen dieser Klasse berechnen lassen, können Greedy-Algorithmen oft näherungsweise Lösungen mit relativ geringer Berechnungs- und Implementierungskomplexität ermitteln.

In diesem Unterabschnitt werden mehrere Greedy-Algorithmen zur positionsbasierten Wegwahl vorgestellt. Der betrachtete ungerichtete Netzwerkgraph sei  $G = (V, E)$ , wobei  $V$  die Menge der Knoten und  $E$  die Menge der Kanten ist. Des Weiteren sei  $s \in V$  der aktuell betrachtete Knoten, von dem aus ein Pfad zu dem Zielknoten  $z \in V$  gefunden werden soll. Alle hier vorgestellten Verfahren gehen davon aus, dass jeder Knoten  $v \in V$  seine eigene Position und die Positionen seiner direkt benachbarten Knoten  $N_v = \{w \in V : (w, v) \in E\}$  kennt. Analog zu AODV wird hierbei die lokale Nachbarschaft der Knoten über den Austausch von Nachrichten via Broadcast-Kommunikation eruiert. Die Positionen des Startknotens  $s$  und des Zielknotens  $z$  sind ebenfalls bekannt, wobei Letztere bei Bedarf mit einem geeigneten dezentralen Verfahren bestimmt wird.

Mithilfe einer Bewertungsfunktion wird nun die Eignung aller direkten Nachbarn des aktuellen Knotens zur Erreichung des Zielknotens ermittelt und anschließend der am besten geeignete Nachbar als nächster Knoten auf dem Pfad zum Zielknoten ausgewählt. Dieser Vorgang wird so lange wiederholt, bis entweder der Zielknoten erreicht oder eine Abbruchbedingung erfüllt wurde. Die Abbruchbedingung ist notwendig, um sicherzustellen, dass bei einem erfolglosen Versuch, einen Pfad zu finden, keine Endlosschleife entsteht. Eine universell einsetzbare Abbruchbedingung ergibt sich beispielsweise durch die Limitierung der Pfadlänge.

Abbildung 2.4 zeigt den Ausschnitt eines Netzwerks, in dem vom Startknoten  $s$  aus der nächste Weiterleiter auf dem Pfad zum Zielknoten  $z$  ermittelt werden soll. Der Knoten  $s$  kennt die Positionen von allen Knoten innerhalb der Funkreichweite, welche als Kreis dargestellt ist, und kann nun gemäß einer Bewertungsfunktion den am besten geeigneten Nachbarn auswählen. *Most Forward within Radius* (MFR) [149] verwendet als Bewertungsfunktion den geografischen Fortschritt in Bezug auf die Strecke  $\overline{sz}$ . Der Knoten  $a$  erzielt von allen Nachbarn den größten Fortschritt und würde entsprechend vom MFR-Verfahren ausgewählt werden. *Greedy Routing Scheme* (GRS) [52] wählt den Nachbarknoten mit dem kleinsten geografischen Abstand zum Zielknoten  $z$  als nächsten Weiterleiter aus. In dem betrachteten Beispiel ist dies der Knoten  $b$ , der in Bezug auf die Strecke  $\overline{sz}$  weniger Fortschritt erzielt als Knoten  $a$ . Um Endlosschleifen zu vermeiden, werden beim GRS-Verfahren im Auswahlprozess nur diejenigen Nachbarn



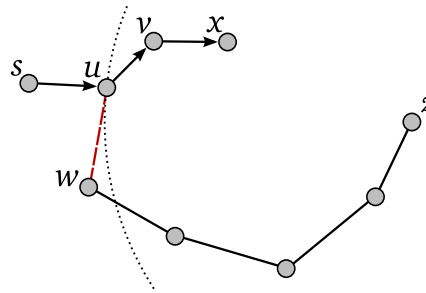
**Abbildung 2.4:** Die Auswahl des am besten geeigneten Weiterleiters anhand unterschiedlicher Bewertungsfunktionen.

berücksichtigt, die einen kleineren Abstand zum Zielknoten aufweisen als der aktuell betrachtete Knoten. Sollte kein Nachbar verfügbar sein, der den Abstand zum Zielknoten verringert, so wird die Suche nach einem Pfad eingestellt. Die Autoren von [147] schlagen vor, die Suche nur einzustellen, wenn der am besten geeignete Nachbar mit dem Vorgängerknoten identisch ist. Durch diese Modifikation ist es möglich, in dem Auswahlprozess auch diejenigen Nachbarn zu berücksichtigen, die weiter vom Ziel entfernt sind als der aktuell betrachtete Knoten. Verfügen die Knoten über die Möglichkeit, ihre Funkreichweite gezielt für jedes Paket zu steuern, so können die Kollisionswahrscheinlichkeit der Übertragungen und ihr Energieverbrauch durch das Verfahren *Nearest Forward Progress* (NFP) [148] reduziert werden. Dieses Verfahren wählt unter allen Nachbarn, die geografischen Fortschritt in Richtung des Zielknotens erzielen, denjenigen aus, der den kleinsten Abstand zum aktuellen Knoten besitzt. Dementsprechend würde NFP in dem Beispiel von Abbildung 2.4 den Knoten *c* als Weiterleiter auswählen. Zwar kann durch diese Strategie die insgesamt benötigte Energiemenge für die Padsuche von *s* nach *z* reduziert werden, gleichzeitig verlängert sich allerdings auch die Pfadlänge beträchtlich.

## Eigenschaften und Limitierungen

Abgesehen von der Position des Zielknotens verwenden alle beschriebenen Greedy-Algorithmen ausschließlich lokales Wissen, um aus den benachbarten Knoten den nächsten Weiterleiter auszuwählen. Da gerade in Einsatzszenarien mit einer hohen Knotenmobilität immer die Gefahr besteht, dass das topologische Wissen über weiter entfernte Regionen nicht mehr aktuell ist, kann die Beschränkung auf lokales Wissen die Stabilität eines Protokolls entscheidend verbessern. Auch ermöglicht diese Limitierung eine im Vergleich zu AODV kompaktere und dadurch wenig fehleranfällige Implementierung.

Sollte allerdings die Zielposition nicht a priori bekannt sein, so muss diese erst mit Hilfe eines dezentralen Verfahrens bestimmt werden. In diesem Fall wird anders als bei AODV ein großer Teil der Protokollkomplexität in ein zweites Verfahren zur Positionsbestimmung ausgelagert.



**Abbildung 2.5:** Die Auswahl des nächsten Weiterleiters basiert bei positionsbasierten Greedy-Algorithmen ausschließlich auf lokalem Wissen und kann deshalb aus globaler Perspektive fehlerhaft sein.

Dennoch ist es weit weniger aufwendig, eine hinreichend genaue Position des Zielknotens zu bestimmen und bei Bedarf zu aktualisieren, als die Stabilität eines langen Pfads innerhalb eines mobilen Netzwerks durchgängig zu gewährleisten. Ein bedeutender Nachteil der rein positionsbasierten Auswahl des Weiterleiters liegt in dem Umstand, dass eine mit lokalem Wissen getroffene Entscheidung aus globaler Sicht nicht zwingend optimal sein muss. Bezogen auf den konkreten Pfadverlauf oder die Pfadlänge ist dieser Umstand meist nur von geringer Bedeutung. Wesentlich gewichtiger ist allerdings der Umstand, dass diese Verfahren aufgrund ihrer lokalen Entscheidungen nicht garantieren können, immer einen Pfad zwischen zwei verbundenen Netzteilnehmern zu finden. In [Abbildung 2.5](#) wird diese Problematik anhand eines Beispiels verdeutlicht. Knoten  $u$  kennt die Positionen der beiden benachbarten Knoten  $v$  und  $w$  und muss entscheiden, welcher besser geeignet ist, um den Knoten  $z$  zu erreichen. Da der Knoten  $v$  näher an  $z$  liegt als  $w$ , wird  $v$  von allen oben beschriebenen Bewertungsfunktionen bevorzugt. Aus globaler Perspektive unter Kenntnis der gesamten Topologie ist ersichtlich, dass in diesem Beispiel der Knoten  $v$  nicht zum Ziel führt. Dieses Problem tritt vorwiegend in Szenarien mit wenigen oder ungleichmäßig verteilten Knoten auf und kann nicht mit lokalem Wissen vermieden werden. Alle positionsbasierten Greedy-Algorithmen sind dementsprechend Heuristiken, die anhand von 1-Hop-Nachbarschaftsbeziehungen einen aus lokaler Sicht geeigneten Weiterleiter auswählen, aber nicht garantieren können, in jedem Fall einen Pfad zu finden.

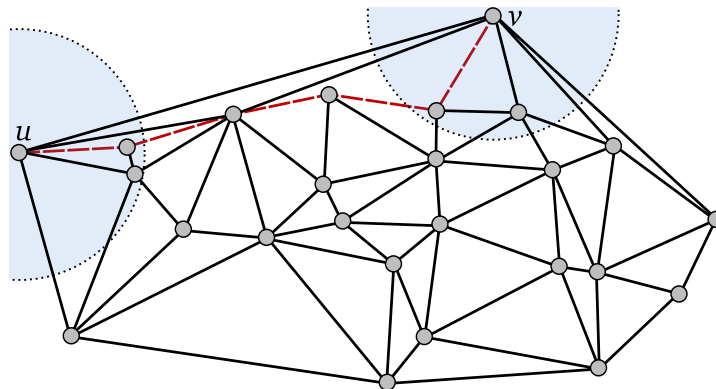
### Compass Routing

*Compass Routing* [[92](#)] ist ein positionsbasierter Greedy-Algorithmus. Er wählt denjenigen Nachbarn  $n \in N_v$  als nächsten Knoten aus, der mit dem Zielknoten  $z$  und dem aktuellen Knoten  $v$  den kleinsten Winkel  $\angle vzn$  formt. Dieser Auswahlvorgang wird so lange wiederholt, bis der Zielknoten erreicht wurde und somit  $v = z$  gilt. Compass Routing wählt dementsprechend im Beispiel von [Abbildung 2.4](#) den Knoten  $d$  als nächsten Weiterleiter aus. Die Besonderheit dieses Ansatzes liegt darin, dass er sich nicht an einem global gültigen Kriterium wie beispielsweise dem Abstand zum Zielknoten orientiert, sondern ein Bewertungsmaß heranzieht, das relativ zum aktuell betrachteten Knoten berechnet wurde. Ist das Bewertungskriterium wie



bei GRS global gültig, so kann der aktuell betrachtete Knoten feststellen, ob einer der zur Auswahl stehenden Nachbarn eine bessere Bewertung erzielt als er selbst. Sollte dies nicht der Fall sein, befindet sich das Verfahren in einem lokalen Optimum und kann beendet werden. Diese Abbruchbedingung stellt sicher, dass der ermittelte Pfad keine Zyklen aufweist. Da die Bewertungsfunktion von Compass Routing immer den Knoten  $n$  mit dem kleinsten Winkel  $\angle vnz$  auswählt, ist das Bezugssystem der Bewertung vom aktuell betrachteten Knoten abhängig. Somit können Zyklen bei der Padsuche in UDGs nicht ausgeschlossen werden.

Allerdings gehen die Autoren im Unterschied zu den zuvor besprochenen Verfahren von einem ungerichteten und planaren Graphen aus. Ein Graph ist dann planar, wenn er auf einer Ebene mit Punkten für die Knoten und Linien für die Kanten überschneidungsfrei dargestellt werden kann. Wie aus der Abbildung 2.1 ersichtlich ist, sind die in Ad-hoc-Netzwerken verwendeten UDGs nicht planar. Da der Compass-Algorithmus auch in einem planaren Graphen nicht für jede Kombination von Start- und Zielknoten einen Pfad findet und auch keine alternative Wegfindung kennt, handelt es sich bei diesem Verfahren wie auch bei den anderen Greedy-Algorithmen um eine Heuristik. Die Wegfindung kann nur garantiert werden, wenn der Netzwerkgraph weitere einschränkende Eigenschaften aufweist. So zeigen die Autoren, dass für Graphen, die aus einer *Delaunay-Triangulation* [44] entstanden sind, die Wegfindung für jede Kombination von Start- und Zielknoten garantiert werden kann. Bei der Delaunay-Triangulation handelt es sich um ein Verfahren, mit dem aus einer Punktemenge ein Dreiecksnetz erstellt wird. Dieses hat die Eigenschaft, dass jeder Kreis, auf dem die drei Eckpunkte eines Dreiecks liegen, keine anderen Punkte enthält. Interpretiert man die gegebenen Netzwerk-knoten als Punktemenge, so kann die Triangulation verwendet werden, um die zugehörigen Kantenmenge  $E$  zu berechnen.



**Abbildung 2.6:** Ein aus der Delaunay-Triangulation entstandener Graph, in dem für die Knoten  $u$  und  $v$  eine exemplarische Radioreichweite eingezeichnet wurde.

In Abbildung 2.6 ist ein aus der Delaunay-Triangulation entstandener Graph zu sehen. Zwar existieren mehrere effiziente Algorithmen wie beispielsweise der Divide-and-Conquer-Algorithmus von Leach [98] oder das Sweep-Verfahren von Fortune [63], welche die Delaunay-

Triangulation in  $\mathcal{O}(n \log n)$  berechnen, allerdings benötigen diese Verfahren die Positionen von allen Knoten. Prinzipiell könnte die Triangulation auch dezentral berechnet werden, indem die notwendigen Informationen zwischen den Knoten ausgetauscht werden. Dieser großflächige Datenaustausch würde aber die verfügbare Bandbreite des Netzwerks reduzieren und gerade bei mobilen Netzwerkknoten das Risiko von Inkonsistenzen mit sich bringen. Wesentlich problematischer ist allerdings der Umstand, dass die Delaunay-Triangulation ausschließlich von den Knotenpositionen abhängt und keine weiteren Parameter wie beispielsweise die Funkreichweite berücksichtigen kann. Der Abbildung 2.6, in der für die Knoten  $u$  und  $v$  die Funkreichweite eingezeichnet wurde, kann entnommen werden, dass die Kantenlänge die Funkreichweite um ein Vielfaches übersteigen kann. In diesem Beispiel müsste die Kante von  $u$  nach  $v$  über einen Pfad mit der Länge von fünf Hops simuliert werden. Der Aufbau dieser zusätzlichen Abstraktionsschicht kann nur durch großflächiges Fluten erreicht werden und könnte dann direkt für einen Distanzvektoralgorithmus verwendet werden. Die Delaunay-Triangulation ist aus diesen Gründen für drahtlose Ad-hoc-Netzwerke ungeeignet. Entsprechend kann nicht garantiert werden, dass Compass Routing in Ad-hoc-Netzwerken immer einen Pfad zum Zielknoten findet.

### Auswahl durch Wettbewerb

Die bisher vorgestellten Greedy-Algorithmen setzen voraus, dass jeder Knoten die Positionen seiner jeweiligen Nachbarn kennt. Hierfür übermittelt er in regelmäßigen Abständen seine eigene Position an alle Knoten innerhalb seiner Funkreichweite. In welchem Intervall diese Übertragung erfolgen muss, hängt maßgeblich von der Mobilität der Knoten ab. Durch diese Kommunikation wird ein beachtlicher Teil der verfügbaren Bandbreite proaktiv für den Aufbau von Nachbarschaftstabellen verwendet, unabhängig davon, ob diese im Augenblick für die Auswahl eines Weiterleiters benötigt werden oder nicht. Als Alternative zu der expliziten Auswahl des nächsten Weiterleiters anhand der übertragenen Positionsdaten kann auch ein verteilter Wettbewerb verwendet werden. Hierfür übermittelt der aktuell betrachtete Knoten das Paket inklusive der Position des Zielknotens an alle Nachbarn. Jeder Empfänger, der sich innerhalb des Weiterleitungsgebiets befindet, bewertet nun anhand der verwendeten Bewertungsfunktion seine eigne Eignung und startet dementsprechend eine Stoppuhr. Beim Einsatz von GRS läuft die Stoppuhr beispielsweise umso schneller ab, je kürzer der Abstand des Knotens zur Zielposition ist. Der Knoten, dessen Stoppuhr zuerst abläuft, liegt in diesem Beispiel dem Ziel am nächsten und leitet das Paket erneut an alle seine Nachbarn weiter. Auf diese Weise werden die anderen Knoten innerhalb des Weiterleitungsgebiets über das Ergebnis des Wettbewerbs informiert und können ihre noch nicht abgelaufenen Stoppuhren abbrechen. Aufgrund dieses Abbruchmechanismus ist das Weiterleitungsgebiet so zu wählen, dass sich alle darin liegenden Knoten gegenseitig hören können. Parallel zum Abbruch der bisher gestarteten Stoppuhren startet die Übertragung des Gewinners gleichzeitig den Wettbewerb um den nächsten Weiterleiter.

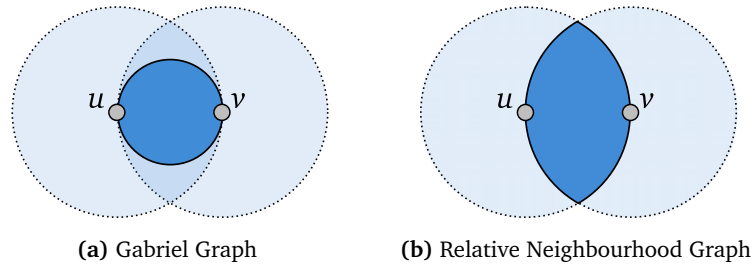
Diese Auswahlstrategie benötigt keine Positionsinformationen der Nachbarknoten, sodass der gesamte proaktive Nachrichtenaustausch entfällt. Daher ist diese Methode besonders in Szenarien mit einer hohen Knotendichte anderen Vorgehensweisen überlegen. Sie hat außerdem den Vorteil, dass die Auswahl durch den verteilten Wettbewerb stets auf den aktuellen Knotenpositionen basiert. Eine Fehlentscheidung aufgrund veralteter Positionsinformationen kann so ausgeschlossen werden. Somit ist diese Auswahlstrategie ebenfalls für Szenarien mit einer hohen Knotenmobilität geeignet. Allerdings sind die Hürden für einen realen Einsatz auch vergleichsweise hoch. Selbst wenn die Geräte über eine hinreichend genaue Uhr verfügen, muss die Implementierung mit dem verwendeten MAC-Protokoll doch schichtübergreifend koordiniert werden. Ansonsten könnten ungewollte zeitliche Verzögerungen während der Wettbewerbsphase auftreten und die Auswahl beeinträchtigen. Da während der Wettbewerbsphasen, die bei jedem Knoten entlang des Pfads jedes Mal erneut stattfinden, keine weiteren Daten geschickt werden können, kann die real erreichte Bandbreite zwischen den Kommunikationspartnern deutlich unter der theoretisch möglichen liegen. Dieser Effekt kann zwar durch eine Verkürzung der Wettbewerbsphase reduziert werden, allerdings steigen hierdurch auch die Anforderungen an die eingesetzte Hardware.

Unter anderem wurden mit *Beaconless Routing* (BLR) [69], *Contention-Based Forwarding* (CBF) [61, 60] und *Implicit Geographic Forwarding* (IGF) [25] zeitnah mehrere auf GRS basierende Wettbewerbsverfahren vorgeschlagen. Ein wesentliches Unterscheidungsmerkmal dieser Verfahren liegt in ihrer Definition des Weiterleitungsgebiets. Eine ausführliche Analyse hierzu kann [36] entnommen werden.

### 2.1.3 Face-2 Routing

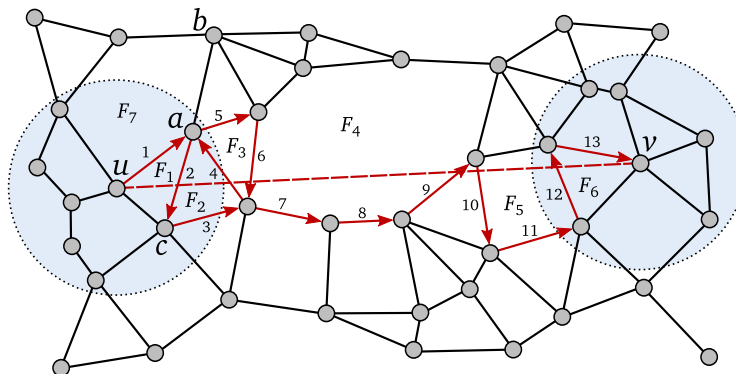
*Face-2 Routing* [27] ist ein positionsbasiertes Routingverfahren, welches garantiert immer einen Pfad zu einem prinzipiell erreichbaren Zielknoten findet. Analog zu den bisher vorgestellten positionsbasierten Ansätzen geht Face-2 Routing davon aus, dass jeder Knoten des betrachteten ungerichteten Netzwerkgraphen  $G = (V, E)$  seine eigene Position und die seiner Nachbarn kennt. Die Idee von Face-2 besteht nun darin, anhand des geometrischen Aufbaus des Graphen einen Pfad zum Zielknoten zu finden. Die Pfadsuche kann allerdings nur für planare Graphen garantiert werden, also solche Graphen, die keine überschneidenden Kanten aufweisen. In Abbildung 2.1 ist zu sehen, dass der Ad-hoc-Netzwerken normalerweise zugrunde liegende UDG überschneidende Kanten enthält und somit nicht planar ist. Allerdings kann der UDG in einen planaren Graphen überführt werden, indem Kanten gelöscht werden.

Der *Gabriel Graph* (GG) [66] und der *Relative Neighbourhood Graph* (RNG) [152] sind zwei bekannte planare Untergraphen der Delaunay-Triangulation. In einem GG sind zwei Knoten  $u$  und  $v \in V$  nur dann miteinander durch eine Kante verbunden, wenn in dem Kreis mit der Strecke  $uv$  als Durchmesser kein anderer Knoten des Graphen liegt. Wie auch schon beim UDG lässt sich auch hier diese Definition direkt verwenden, um aus einer Knotenmenge die zugehörige Kantenmenge zu generieren. Analog zum GG kann auch der RNG direkt aus der Knotenmenge generiert werden. Hierfür werden die zwei Knoten  $u$  und  $v \in V$  immer dann



**Abbildung 2.7:** Knoten  $u$  und  $v$  werden genau dann über eine Kante verbunden, wenn sich kein anderer Knoten in dem dunkelblau gefärbten Gebiet befindet.

miteinander verbunden, wenn kein Knoten  $w \in V$  existiert, der einen kleineren Abstand zu  $u$  und  $v$  aufweist als diese untereinander. In Abbildung 2.7 werden die beschriebenen Konstruktionsbedingungen des GG und des RNG dargestellt. Die beiden in der Abbildung dargestellten Knoten werden nur dann durch eine Kante miteinander verbunden, wenn sich kein anderer Knoten in dem dunkelblau gefärbten Gebiet befindet. Sowohl der RNG als auch der GG beinhalten den minimalen Spannbaum der Delaunay-Triangulation [118]. Auf diesem Sachverhalt basierend zeigt [27], dass die Kantenschnittmenge des UDG und des GG zu einem verbundenen planaren Graphen führt. Da Face-2 nur von der Kenntnis der lokalen Nachbarschaft ausgeht und somit der gesamte Graph für den einzelnen Knoten unbekannt ist, kann jeder Knoten den lokalen Ausschnitt des GG oder des RNG bestimmen, indem er ungültige Kanten zu seinen Nachbarn löscht. Dementsprechend werden in einer realen Protokollimplementierung die Nachbarn ohne gültige virtuelle Kante bei der Auswahl des nächsten Weiterleiters ignoriert. Wird die Kantenmenge des UDG aus Abbildung 2.1 mit der Kantenmenge des entsprechenden GG geschnitten, so entsteht der in Abbildung 2.8 dargestellte planare Graph.



**Abbildung 2.8:** Ein durch die Kantenschnittmenge des UDG und des GG entstandener planarer Graph, in dem der von Face-2 gewählte Pfad rot eingezeichnet ist.

Jeder planare Graph setzt sich aus räumlich zusammenhängenden Gebieten zusammen, die *Faces* genannt werden. Mit diesem Begriff werden nicht nur die räumlich geschlossenen Gebiete

im Netzwerkkinnern bezeichnet, sondern auch die offenen Bereiche an den Netzwerkgrenzen. Dementsprechend ist jede Kante des Graphen Bestandteil von zwei Faces. Wird eine Kante von Knoten  $u$  zu Knoten  $a$  besucht, so liegt sowohl links als auch rechts von der Kante  $(u, a)$  ein Face. In Abbildung 2.8 sind diese beiden Faces mit  $F_7$  und  $F_1$  beschriftet. Es ist nun möglich, von der Kante  $(u, a)$  ausgehend einen Pfad entlang der Grenze von Face  $F_1$  zu beschreiten; dabei wird immer diejenige Kante zu einem benachbarten Knoten ausgewählt, die entgegen dem Uhrzeigersinn den kleinsten Winkel zu der zuvor besuchten Kante aufweist. Analog kann die Grenze von  $F_7$  besucht werden, indem stets die Kante mit dem kleinsten Winkel im Uhrzeigersinn ausgewählt wird. Diese Strategie basiert auf der bekannten *Rechte-Hand-Regel* [26], die verwendet wird, um einen Ausgang aus einem Irrgarten zu finden. Die Idee von Face-2 besteht nun darin, bei der Pfadsuche von Knoten  $u$  nach Knoten  $v$  die von der Strecke  $\overline{uv}$  geschnittenen Faces so lange nacheinander zu besuchen, bis schließlich der Knoten  $v$  erreicht wurde.

Zu Beginn wird derjenige Nachbar von Knoten  $u$  ausgewählt, dessen zugehörige Kante den kleinsten Winkel zu dem Vektor  $\vec{vu}$  aufweist. In dem betrachteten Beispiel ist dies der Knoten  $a$ . Um die Grenze des gewünschten Faces zu besuchen, ist es notwendig, die entsprechende Rotationsrichtung während der Pfadsuche im Paketkopf mitzuführen. Entsprechend wird bei dem zuerst ausgewählten Knoten getestet, ob der kleinere Winkel im Uhrzeigersinn liegt oder nicht. Hierbei ist insbesondere die Richtung des Vektors  $\vec{vu}$  zu beachten. Im Beispiel wird ausgehend vom dem Vektor  $\vec{vu}$  der kleinere Winkel zu der Kante  $(u, a)$  entgegen dem Uhrzeigersinn gemessen. Deshalb wird als anfängliche Rotationsrichtung der Pfadsuche ebenfalls die Richtung gegen den Uhrzeigersinn ausgewählt. Der Grenze des aktuellen Faces wird so lange gefolgt, bis die zuletzt besuchte Kante von Knoten  $v_1$  zu  $v_2$  den Vektor  $\vec{vu}$  schneidet. Liegt der Zielknoten rechts von dem Vektor  $\overline{v_1v_2}$ , wird die Richtung gegen den Uhrzeigersinn als Rotationsrichtung ausgewählt. Sollte der Zielknoten links vom Vektor liegen, so wird der Uhrzeigersinn als Rotationsrichtung verwendet. Gemäß der ausgewählten Rotationsrichtung wird das entsprechende nächste Face wieder so lange durchlaufen, bis die zuletzt besuchte Kante sich mit  $\vec{vu}$  schneidet. Diese Schritte werden wiederholt, bis der Zielknoten erreicht ist. In Abbildung 2.8 sind die während der Pfadsuche besuchten Kanten rot hervorgehoben und der Reihenfolge nach nummeriert. Die hier beschriebene Pfadsuche orientiert sich an der Darstellung aus [43], die im Vergleich zur Erstpublikation wesentlich detaillierter und praxisnäher ausfällt. Die mathematische Darstellung der Erstpublikation ist vermutlich auch die Hauptursache dafür, dass kurz nach deren Erscheinen ein nahezu identisches, aber stärker beachtetes Verfahren mit dem Namen *Perimeter Routing* in [88] vorgeschlagen wurde.

Die bisher vorgestellten positionsbasierten Routingverfahren konnten nicht garantieren, dass sie einen Pfad zu jedem vom Startknoten erreichbaren Knoten finden. Die einzige Ausnahme hierbei bildet Compass Routing, das aber eine für Ad-hoc-Netzwerke unrealistische Delaunay-Triangulation voraussetzt. Auch Face-2 stellt für eine garantierte Pfadsuche Vorbedingungen an den Netzwerkgraphen, zeigt aber mit den beschriebenen Verfahren eine realistische Möglichkeit auf, wie der Graph dezentral planarisiert werden kann. Ein Vorteil gegenüber AODV besteht in der Tatsache, dass die Pfadsuche ohne Paketduplizierung und Fluten

auskommt. Allerdings sind die gefundenen Pfade vergleichsweise lang. Sind  $c$  die Kosten für den kostengünstigsten Pfad zwischen einem Knotenpaar, so können die Kosten für ein positionsbasiertes Verfahren mit lokalem Wissen bis auf  $\Omega(c^2)$  ansteigen [95]. Hierfür können die Anzahl der besuchten Kanten, die euklidische Länge des Pfads oder die für die Pfadsuche eingesetzte Energiemenge als Kostenkriterien herangezogen werden. Diese Limitierung gilt auch für die positionsbasierten Greedy-Algorithmen, welche aber die Pfadsuche bei Bedarf abbrechen und somit die durchschnittlichen Kosten der gefundenen Pfade senken können.

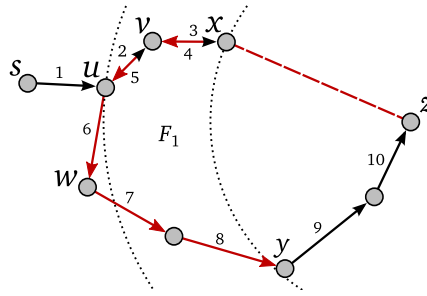
### 2.1.4 Kombinationsmöglichkeiten verschiedener Verfahren

GRS ist eine der am häufigsten eingesetzten Bewertungsfunktionen bei positionsbasierten Greedy-Algorithmen. Ihr gemäß wird immer derjenige Nachbar als nächster Weiterleiter ausgewählt, der den kleinsten euklidischen Abstand zum Zielknoten aufweist. Durch den hohen Bekanntheitsgrad wird dieses Verfahren in zahlreichen Publikationen einfach nur Greedy Routing genannt. Zwar kann der Erfolg einer Pfadsuche mittels GRS nicht garantiert werden, allerdings ist die Wahrscheinlichkeit, mit diesem Verfahren einen Pfad zu finden, für viele Einsatzzwecke ausreichend hoch. Dennoch führt dieser Umstand dazu, dass GRS kein universell einsetzbares Verfahren ist. In diesem Unterabschnitt werden deshalb Möglichkeiten dargestellt, GRS mit anderen Ansätzen zu kombinieren.

#### Greedy-Face-Greedy

Sollte sich kein Nachbar näher am Zielknoten befinden als der aktuell betrachtete Knoten, so stellt GRS die Pfadsuche ein. Die Idee von *Greedy-Face-Greedy* (GFG) [27] besteht nun darin, die von GRS abgebrochene Pfadsuche mit Hilfe des Face-2-Algorithmus fortzusetzen. Abbildung 2.9 veranschaulicht diese Strategie anhand eines Beispiels. Ausgehend vom Knoten  $u$  versucht GRS, einen Pfad zu Knoten  $v$  zu finden, stellt die Suche aber bei Knoten  $x$  erfolglos ein. Nun wird von Knoten  $x$  aus die Suche mit Hilfe von Face-2 so lange fortgeführt, bis der Zielknoten  $v$  oder ein Knoten  $y$  mit  $d(y, v) < d(x, v)$  erreicht wird. In dem Beispiel wechselt GFG bei Knoten  $y$  wieder zu GRS und erreicht schließlich das Ziel. Durch diese Kombination kann zum einen garantiert werden, dass ein Pfad zum Zielknoten gefunden wird, zum anderen ist die Pfadlänge im Durchschnitt deutlich kürzer als bei dem ausschließlichen Einsatz von Face-2. Da es sich bei beiden Routingverfahren um positionsbasierte Ansätze handelt, ist die Zielposition bei dem Einsatz von Face-2 ohnehin bekannt. Somit kann zwischen beiden Verfahren ohne zusätzlichen Aufwand und ohne Fluten gewechselt werden.

Diese Kombination wird auch in [88] unter dem Namen *Greedy Perimeter Stateless Routing* (GPSR) vorgeschlagen und auf Paketebene simuliert. In [94] wird die Kombination von GRS und einer eigenen Variante des Face-2-Algorithmus untersucht. Analog zum vorherigen Unterabschnitt bezeichnet  $c$  die Kosten für den kostengünstigsten Pfad zwischen einem Knotenpaar. Die Autoren von [94] zeigen, dass die beschriebene, GOAFR+ genannte Kombination asymptotisch optimal ist und die Kosten der gefundenen Pfade auf  $\mathcal{O}(c^2)$  begrenzt sind.



**Abbildung 2.9:** Die Pfadsuche von GRS befindet sich bei Knoten  $x$  in einem lokalen Optimum, das durch den ergänzenden Einsatz von Face-2 überbrückt wird.

Es wurden mehrere Varianten von GRS vorgeschlagen, die auf den expliziten Austausch von Positionsdaten verzichten und stattdessen einen verteilten Wettbewerb zur Auswahl des nächsten Weiterleiters verwenden. Da Face-2 aber die Positionen der Nachbarn benötigt, kann dieses Verfahren nicht ohne Weiteres mit den Wettbewerbsvarianten von GRS kombiniert werden. Die Autoren von *Beaconless Routing* (BLR) schlagen deshalb in [70] vor, die Positionsdaten während des Einsatzes von Face-2 von den Nachbarn explizit zu erfragen und dann zur Planarisierung zu verwenden. Da jeder Knoten in der Nachbarschaft auf die Anfrage antworten muss, ist die Anzahl der auf diese Weise ausgetauschten Nachrichten vergleichsweise hoch. [35] formuliert die Idee, den planaren GG durch einen verteilten Wettbewerb zu konstruieren und somit die Nachrichtenmenge wesentlich zu reduzieren. *Beaconless Forwarder Planarization* (BFP) [130] verallgemeinert dieses Funktionsprinzip und ermöglicht die verteilte Bestimmung von zahlreichen weiteren planaren Graphen wie beispielsweise des RNG. *No-Beacon Face* (NB-Face) [112] kombiniert die Konstruktion des planaren Graphen direkt mit der Auswahl der nächsten Kante des aktuell besuchten Faces und kann hierdurch die Nachrichtenmenge weiter reduzieren. Diese Strategie wird in [130] und [131] im Detail analysiert und weiter verbessert.

## Position-Based Routing with Distance Vector Recovery

Anstatt GRS mit einem positionsbasierten Ansatz zu kombinieren, besteht auch die Möglichkeit, ein topologiebasiertes Verfahren zur Überwindung des lokalen Optimums heranzuziehen. [56] und [138] schlagen hierfür den Einsatz von AODV mit einer modifizierten Pfadsuche vor. In den RREQ-Paketen werden zusätzlich die Positionen des zuletzt von GRS besuchten Knotens sowie des Zielknotens mitgeführt. In dem Beispiel in Abbildung 2.9 entspricht dies den Positionen von  $x$  und  $z$ . Empfängt ein Knoten ein RREQ-Paket, so vergleicht er seinen eigenen Abstand zum Zielknoten mit demjenigen des im Paketkopf hinterlegten Knotens. Sollte der Empfänger näher am Zielknoten liegen oder gar selbst der Zielknoten sein, so sendet er entlang des Hinwegs ein RREP-Paket zum Initiator der topologischen Pfadsuche. Zwar verwendet diese Strategie Fluten, um einen Überbrückungspfad zu finden, dafür sind aber die entdeckten Pfade im Durchschnitt wesentlich kürzer als bei den Varianten mit Face-2.



## Greedy Perimeter Coordinator Routing

Die Autoren von GPSR haben in [87] darauf hingewiesen, dass die dezentrale Berechnung des GG durch Funkhindernisse beeinträchtigt werden kann. Eine fehlerhafte Planarisierung kann wiederum zu Fehlentscheidungen während der Pfadsuche von Face-2 führen. Aufgrund der hohen Bebauungsdichte in Städten kann deshalb davon ausgegangen werden, dass beim Einsatz von Face-2 zur Fahrzeug-zu-Fahrzeug-Kommunikation erhebliche Probleme auftreten können. Aus diesem Grund kombiniert *Greedy Perimeter Coordinator Routing* (GPCR) [103, 101] GRS und Face-2 auf eine speziell an Innenstädte angepasste Weise miteinander. Diesem Verfahren liegt die Beobachtung zugrunde, dass die wichtigen Routingentscheidungen an den Straßenkreuzungen getroffen werden müssen. Anders als die im Unterabschnitt 2.1.5 beschriebenen Verfahren versucht GPCR jedoch, diese Kreuzungen ohne digitale Karte zu erkennen.

Hierfür werden zwei Verfahren vorgeschlagen. Das erste von ihnen benötigt eine 2-Hop-Nachbarschaftstabelle, für die jeder Knoten sowohl seine eigene Position als auch die seiner Nachbarn in regelmäßigen Abständen an alle Knoten innerhalb seiner Funkreichweite übertragen muss. Sollte ein Knoten  $c$  zwei Nachbarn kennen, deren geografischer Abstand zueinander kleiner ist als die Funkreichweite, die sich aber nicht gegenseitig in den von ihnen übertragenen Nachbarschaftslisten nennen, so befindet sich der Knoten  $c$  vermutlich auf einer Kreuzung und die beiden Nachbarn sind durch ein Gebäude voneinander getrennt. Für das zweite Verfahren genügt die Kenntnis der direkten Nachbarschaft, die von GRS und Face-2 ohnehin vorausgesetzt wird. Jeder Knoten berechnet anhand der  $x$ - und  $y$ -Koordinaten seiner Nachbarn den Korrelationskoeffizienten:

$$p_{xy} = \left| \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2 \sum_{i=1}^n (y_i - \bar{y})^2}} \right| \quad (2.1)$$

Ist  $p_{xy} \in [0, 1]$  nur geringfügig kleiner als eins, so sind die Positionen der Nachbarn linear voneinander abhängig. Dementsprechend befindet sich der betrachtete Knoten wahrscheinlich nicht auf einer Kreuzung. Die Autoren schlagen vor, bei allen Knoten mit  $p_{xy} < 0,9$  davon auszugehen, dass sie sich auf einer Kreuzung befinden.

Jeder Knoten schickt in regelmäßigen Abständen seinen Identifikator und seine aktuelle Position an alle Knoten innerhalb seiner Funkreichweite. Bei GPCR wird zusätzlich übermittelt, ob der Knoten davon ausgeht, auf einer Kreuzung zu sein. GPCR verwendet GRS als Basisstrategie und wählt dementsprechend denjenigen Nachbarn als nächsten Weiterleiter aus, der den kleinsten Abstand zum Zielknoten aufweist. Befindet sich aber ein Nachbar mit geografischem Fortschritt in Richtung des Zielknotens auf einer Straßenkreuzung, so wird dieser im Auswahlprozess bevorzugt. Diese Strategie wird *Restricted Greedy Routing* (RGR) genannt. Sollte während der Pfadsuche bei Knoten  $x$  ein lokales Optimum auftreten, wechselt RGR in einen auf Face-2 basierenden Überbrückungsmodus. Hierbei wird das Paket gemäß der Rechte-Hand-Regel von Straßenkreuzung zu Straßenkreuzung weitergeleitet, bis entweder der Zielknoten  $v$  oder aber ein Knoten  $y$  mit  $d(y, v) < d(x, v)$  erreicht wird. In diesem Überbrückungsmodus



wird ebenfalls RGR eingesetzt, um das Paket entlang der ausgewählten Straßen weiterzuleiten. Sobald ein Knoten gefunden wird, der näher am Zielknoten liegt als der Knoten im lokalen Optimum, wird die Überbrückung beendet und das Paket wird wieder mit RGR direkt in Richtung des Zielknotens weitergeleitet.

Die Eignung von GPCR wurde in einer Simulationsstudie untersucht. Hierbei konnte das Verfahren eine höhere Paketzustellrate erzielen als GPSR. Leider ist das verwendete Simulationsszenario zu klein, um verlässliche Aussagen über die Zuverlässigkeit der Überbrückungsstrategie zu machen.

### 2.1.5 Verfahren zur Wegewahl mit globalem Zusatzwissen

Gerade im Bereich der Fahrzeug-zu-Fahrzeug-Kommunikation ist es naheliegend, davon auszugehen, dass die Fahrzeuge über ein Navigationssystem und somit über eine digitale Karte verfügen. Diese kann als planarer Graph interpretiert und bei der Pfadsuche genutzt werden.

#### Geographic Source Routing

*Geographic Source Routing* (GSR) [104, 23] geht wie die anderen positionsbasierten Verfahren davon aus, dass die Position des Zielknotens bekannt ist oder zu Beginn der Suche bestimmt wird. Mit Hilfe der digitalen Karte wird nun ermittelt, auf welchem Straßenabschnitt sich der Zielknoten befindet. Anschließend berechnet GSR den kürzesten Pfad zwischen der aktuell befahrenen Straße und derjenigen des Zielknotens. Dieser Pfad, der als Liste von Straßenkreuzungen vorliegt, wird in jedes an den Zielknoten adressierte Datenpaket eingetragen. Des Weiteren trägt der Startknoten die erste Straßenkreuzung als Zieladresse in das Paket ein und leitet es per GRS an einen Knoten auf der ausgewählten Straßenkreuzung weiter. Solange noch eine Straßenkreuzung im Paketkopf hinterlegt ist, wird diese vom Empfänger als nächstes Ziel eingetragen und das Paket erneut per GRS verschickt. Nachdem die letzte Straßenkreuzung erreicht wurde, wird das Paket schließlich an den Zielknoten weitergeleitet.

Die Autoren simulierten und evaluierten GSR in einem Innenstadtszenario mit realistischem Verkehrsfluss. Zur Bestimmung der Zielpositionen wurde der *Reactive Location Service* (RLS) [93] verwendet. In der Simulation konnte GSR im Vergleich zu AODV und DSR überzeugende Ergebnisse erzielen. Allerdings war die Fahrzeugdichte in dem gewählten Szenario recht hoch, sodass vermutlich durchgehend von jeder Straßenkreuzung aus jede angrenzende Straßenkreuzung per GSR erreichbar war. In einem realen Einsatz wäre aber damit zu rechnen, dass nicht zu jedem Zeitpunkt jede Straße in ausreichendem Maße von entsprechend ausgerüsteten Fahrzeugen befahren wird. Wie in einem solchen Fall zu verfahren wäre, lässt die Beschreibung von GSR offen.

#### Street Topology Based Routing

*Street Topology Based Routing* (STBR) [64] verwendet wie GSR eine digitale Karte zur Paketvermittlung. Jede Straßenkreuzung wird als Knoten und jede Straße als Kante eines planaren

Graphen interpretiert. In der Umgebung jeder Kreuzung wird ein Knoten ausgewählt, der die Rolle eines Vermittlers übernimmt. Jeder dieser Vermittler kommuniziert per CBF [61, 60] mit den Vermittlern auf den benachbarten Straßenkreuzungen. Hierdurch wird festgestellt, ob sich ausreichend viele Fahrzeuge auf der Straße befinden, damit diese für die Paketvermittlung verwendet werden kann. Des Weiteren wird der kontinuierliche Datenaustausch zwischen den Vermittlern auf den Straßenkreuzungen genutzt, um eine 2-Hop-Nachbarschaftstabelle aufzubauen. Zu Beginn der Pfadsuche leitet der Startknoten das Paket per CBF an den Vermittler der nächstliegenden Straßenkreuzung weiter. Von dort aus wird das Paket gemäß der 2-Hop-Nachbarschaftstabelle an diejenige Straßenkreuzung weitergeleitet, die den kleinsten euklidischen Abstand zum Zielknoten aufweist. Dieser Schritt wird so lange wiederholt, bis der Zielknoten über eine angrenzende Straße erreicht werden kann.

STBR ist ein vergleichsweise komplexes Protokoll mit vielen Herausforderungen. Da sich in den meisten Fällen ein Fahrzeug nur kurzzeitig in der Nähe einer Kreuzung aufhält, ist insbesondere der Auswahlprozess der Vermittlerknoten fehleranfällig. Auch müssen zahlreiche Sonderfälle berücksichtigt werden; so ist es zum Beispiel möglich, dass keine Kreuzung mit Fortschritt zum Zielknoten bekannt ist.

## 2.2 Kooperative Anwendungen

Die bisher vorgestellten Routingverfahren können eingesetzt werden, um Daten zwischen zwei nicht benachbarten Knoten auszutauschen. Von den weiterleitenden Knoten werden hierfür die Nutzdaten zum jeweils nächsten vom Routingprotokoll ausgewählten Nachbarknoten übertragen. Da diese Weiterleiter als reine Vermittler dienen und vom Absender nicht explizit adressiert werden, werden die Nutzdaten auf diesen Knoten nicht zur Anwendungsebene hochgereicht. Des Weiteren ist diesen Knoten auch nicht zwangsläufig der Aufbau der transportierten Nutzdaten bekannt. Der Austausch von Daten und deren Interpretation obliegt somit ausschließlich den beiden Kommunikationspartnern. Diese Übermittlungsform wird Ende-zu-Ende-Kommunikation genannt und ist in den meisten drahtgebunden Netzwerken vorherrschend. Mit Hilfe eines geeigneten Routingprotokolls können daher viele der bestehenden Anwendungen auch in einem Ad-hoc-Netzwerk verwendet werden.

Neben dieser klassischen Kommunikationsform gibt es aber auch andere Ansätze, bei denen die Netzwerkknoten versuchen, kooperativ eine Lösung für eine vorgegebene Problemstellung zu finden. Dabei steht nicht die Kommunikation zwischen zwei explizit adressierten Knoten im Vordergrund, sondern der kooperative Datenaustausch zwischen benachbarten Knoten. In einem solchen Szenario sind die ausgetauschten Daten entweder für alle teilnehmenden Knoten bestimmt, oder die Adressierung der gewünschten Empfänger erfolgt primär anhand der im Paket enthaltenen Nutzdaten. In den folgenden beiden Abschnitten werden exemplarisch derartige Anwendungen aus dem Bereich der Sensornetze und der Fahrzeug-zu-Fahrzeug-Kommunikation präsentiert.

### 2.2.1 Anwendungen in Sensornetzwerken

Sensornetzwerke sind Ansammlungen sehr vieler Kleinstrechner, die mit einem Radiofunksystem und mehreren Sensoren ausgestattet sind. Diese sogenannten Sensorknoten sind im Regelfall batteriebetrieben und verfügen nur über eine sehr begrenzte Speicher- und Rechenkapazität. Der primäre Zweck eines Sensornetzwerks besteht normalerweise darin, eine bestimmte Region mit Hilfe der Sensoren zu überwachen. Es handelt sich also um geschlossene Systeme, die meist nur über einen Knoten – die sogenannte Datensenke – mit der Außenwelt verbunden sind.

Nachdem die Datensenke den betroffenen Knoten mitgeteilt hat, welche Sensordaten von Interesse sind, werden diese kontinuierlich oder bei Eintritt eines vordefinierten Ereignisses – zum Beispiel einer Bewegung oder dem Überschreiten eines bestimmten Schwellenwerts – von den jeweiligen Knoten zur Datensenke weitergeleitet. Um die Belastung und den Energieverbrauch der Datensenke und der umliegenden Knoten zu reduzieren, werden die Sensordaten meist nicht mittels eines Routingverfahrens in Rohform an die Datensenke weitergeleitet, sondern schon während des Weiterleitens innerhalb des Netzwerks aggregiert. Wenn beispielsweise ein Knoten eine Bewegung erkannt hat und die entsprechenden Daten weiterleitet, kann ein benachbarter Weiterleiter die Genauigkeit des geschätzten Aufenthaltsorts verbessern, indem er seine eigenen Beobachtungen mit denen des anderen Knotens kombiniert und nur das Ergebnis in Richtung der Datensenke weitertransportiert. Hierfür kann beispielsweise *Directed Diffusion* [81] oder *COUGAR* [165] verwendet werden.

Andere Ansätze wie beispielsweise *TAG* [107] oder *TiNA* [145] interpretieren das gesamte Sensornetzwerk als verteilte Datenbank, auf die mit einer an SQL angelehnten Abfragesprache zugegriffen werden kann. Auch wenn sich die bisher genannten Verfahren im Detail unterscheiden, so besteht doch ihre grundsätzliche Strategie darin, das Netzwerk zu Beginn zu fluten und einen Spannbaum mit der Datensenke als Wurzel aufzubauen. Die Abfragen werden dann entlang dieses Spannbaums an alle betroffenen Sensorknoten weitergeleitet. Die Blätter des Spannbaums senden die gewünschten Daten an ihren Vater, der die Werte der Söhne sowie seine eigenen aggregiert und das Ergebnis wiederum zu seinem Vater überträgt, bis schließlich die Wurzel erreicht ist.

Sensorknoten besitzen meist nur einen begrenzten Energievorrat, sodass ihre Lebensdauer primär von der Anzahl der übertragenen Pakete abhängt. Um aus diesem Energiemangel resultierenden Ausfällen entgegenzuwirken, kann beispielsweise auf die in [113] beschriebene Mehrwegeweiterleitung zurückgegriffen werden. Sowohl die Nichtberücksichtigung eines Teilbaums aufgrund ausgefallener Knoten als auch die mehrfache Berücksichtigung von Sensorwerten aufgrund einer Mehrwegeweiterleitung kann zu Aggregierungsfehlern führen. Aus diesem Grund wird in [38] vorgeschlagen, für den Datenaustausch und die dezentrale Bestimmung von einfachen Aggregierungsfunktionen wie COUNT oder SUM eine probabilistische Datenstruktur zu verwenden. Zwar kann hierdurch das Ergebnis nur näherungsweise bestimmt werden, dafür ist allerdings die Datenstruktur nicht anfällig gegenüber Dubletten und benötigt vergleichsweise wenig Speicherplatz.

Auf den Basisverfahren aufbauend wurden zahlreiche Anwendungen entwickelt. Beispielsweise wurden im Rahmen des *Great-Duck-Island-Projekts* [108] das Brutverhalten von Wellenläufern und die genauen klimatischen Bedingungen in ihren Nestern untersucht. Hierfür wurden 32 Sensorknoten in und um die Nester positioniert und über Satellit mit dem Internet verbunden. So konnten die Sensordaten in Echtzeit ausgewertet werden, ohne den Lebensraum der Vögel zu beeinträchtigen. Das Netzwerk wurde mehrfach vergrößert und umfasste 2003 mehr als 150 Sensorknoten. Die Kommunikation mit der Basisstation, die mit dem Internet verbunden war, erfolgte durch Multi-Hop Routing.

Eine weitere bekannte Anwendung ist das *ZebraNet* der Universität Princeton [84]. Im Gegensatz zum Great-Duck-Island-Projekt stand hierbei nicht die Überwachung der Umgebung der Tiere im Fokus, sondern die Aufzeichnung ihrer Bewegungsmuster in einem großen Naturschutzgebiet. Die Tiere wurden deshalb mit einem Halsband ausgestattet, das einen GPS-Empfänger und einen Sensorknoten mit zwei Funkeinheiten enthielt. Mit Hilfe der Kurzstreckeneinheit wurden die lokalen Sensordaten zu anderen Knoten innerhalb der Funkreichweite übertragen. Die zweite Funkeinheit verfügte über eine deutliche größere Reichweite und diente dem Datenaustausch mit der Basisstation. Später folgten weitere Anwendungen wie der *Electronic Shepherd* [150] oder ein Projekt zur Überwachung der Aga-Kröte [73]. Neben Anwendungen zur Tierüberwachung gibt es zahlreiche weitere Projekte zur Überwachung gefährlicher Regionen. So wurde beispielsweise in [158] die seismische Aktivität des Tungurahua-Vulkans mit Hilfe von Sensorknoten aufgezeichnet und überwacht.

Alle beschriebenen Verfahren und Anwendungen dienen dazu, ein geografisches Gebiet oder eine Gruppe von Lebewesen mit Hilfe von Sensorknoten zu überwachen. Entweder schicken diese regelmäßig oder ereignisbasiert ihre Daten an eine mit der Außenwelt verbundene Datensinke, oder sie beantworten dezentral deren gezielte Datenabfragen. In beiden Fällen werden die Daten meist dezentral aggregiert und direkt zwischen den Knoten ausgetauscht. Somit versuchen alle Knoten, kooperativ eine bestimmte Aufgabenstellung zu lösen, ohne dabei auf eine explizite Vermittlungsschicht angewiesen zu sein.

### 2.2.2 Anwendungen in Fahrzeug-ad-hoc-Netzwerken

Die grundsätzliche Intention der Fahrzeug-zu-Fahrzeug-Kommunikation besteht in der Regel darin, die Sicherheit der Verkehrsteilnehmer zu verbessern. Falls beispielsweise die Traktionskontrolle in einem Fahrzeug bemerkt, dass die Räder anfangen durchzudrehen und gleichzeitig die Temperatur unter dem Gefrierpunkt liegt, so könnte ein Kommunikationssystem die nachfolgenden Autos vor möglicherweise vorhandenem Glatteis warnen. Gleiches gilt bei plötzlichen Bremsaktionen oder anderen sicherheitsrelevanten Beobachtungen. Die Fahrzeuge können diese Informationen mit eigenen lokalen Beobachtungen kombinieren und im Bedarfsfall weiterleiten, um auf diese Weise auch weiter entfernte Autos zu informieren, deren Fahrer das Gefahrengebiet möglicherweise noch nicht einsehen können.

Eines der ersten Forschungsprojekte in Deutschland, das sich mit dieser Thematik auseinandergesetzt hat, war das *FleetNet-Projekt* [48]. Neben Anwendungen zur unmittelbaren Stei-

gerung der Sicherheit von Verkehrsteilnehmern wurde auch ein dezentral organisiertes Verkehrsinformationssystem entwickelt. Die Idee des *Self-Organizing Traffic Information System* (SOTIS) [162, 160] genannten Ansatzes besteht darin, Fahrzeuge mit GPS, einem digitalen Radio und einer digitalen Karte auszustatten und sie dann ad hoc aktuelle Verkehrsflussinformationen austauschen zu lassen. Dafür werden die Straßen der digitalen Karte in kürzere Abschnitte unterteilt, wobei für beide Fahrtrichtungen eines Abschnitts jeweils ein eindeutiger Identifikator festgelegt wird. Die Länge der Abschnitte wird so gewählt, dass der Abstand zwischen zwei Positionen innerhalb eines Abschnitts immer kleiner ist als die verwendete Funkreichweite. In den durchgeführten Simulationen wurde beispielsweise eine Funkreichweite von 1000 m verwendet, und die Länge der Straßenabschnitte wurde auf 500 m begrenzt. In [161] wird eine alternative Möglichkeit zur Aufteilung solcher Straßen beschrieben, für die keine gemeinsame digitale Karte erforderlich ist.

Um die Verkehrssituation zu bestimmen, überträgt jedes Fahrzeug in regelmäßigen Abständen per Broadcast an alle anderen Fahrzeuge innerhalb der Funkreichweite unter anderem seine Geschwindigkeit und den Identifikator des Straßenabschnitts, auf dem es sich gerade befindet. Auf diese Weise kann jedes Fahrzeug auf mindestens zwei Straßenabschnitten – dem aktuell befahrenen und dem der entgegengesetzten Fahrtrichtung – die aktuelle Verkehrssituation abschätzen. Da davon auszugehen ist, dass nicht alle Verkehrsteilnehmer mit einem solchen System ausgestattet sind, wird die Verkehrssituation nicht anhand der Anzahl der Fahrzeuge auf einem Straßenabschnitt bewertet, sondern mit Hilfe der Durchschnittsgeschwindigkeit. Neben den zuvor aufgeführten Informationen sendet jedes Fahrzeug auch eine Liste mit bereits aggregierten Informationen. Jeder Eintrag in dieser Liste enthält den Identifikator des jeweiligen Abschnitts, die dort beobachtete Durchschnittsgeschwindigkeit und den Zeitpunkt der Beobachtung beziehungsweise Berechnung. Sowohl die eigenen Beobachtungen als auch die von anderen Knoten empfangenen Aggregate werden in einer Datenbank gespeichert. Welche Einträge für die regelmäßig übertragene Liste ausgewählt werden, hängt unter anderem vom Abstand zwischen dem aktuellen Aufenthaltsort und dem Ort der Beobachtung ab. Überschreitet ein Eintrag ein festgelegtes Alter oder bezieht sich auf eine zu weit entfernte Position, wird er aus der Datenbank gelöscht.

Auch *TrafficView* [110, 42] hat das Ziel, dem Fahrer Informationen über die Verkehrssituation auf weiter entfernten Straßenabschnitten bereitzustellen. Im Unterschied zu SOTIS wird hier allerdings nicht davon ausgegangen, dass alle Teilnehmer eine gemeinsame digitale Karte besitzen. Anstatt die Geschwindigkeitsdaten aller Fahrzeuge eines Straßenabschnitts miteinander zu kombinieren und somit gewissermaßen schon eine Analyse der Verkehrssituation zu erstellen, dient die Aggregation bei *TrafficView* vorrangig der Datenreduktion. Für diesen Zweck werden die Daten von Fahrzeugen mit ähnlichen Positions- und Geschwindigkeitsinformationen miteinander kombiniert. Die so erzeugten Datensätze enthalten den gewichteten Durchschnitt der Positionen und Geschwindigkeiten der jeweiligen Fahrzeuge. Des Weiteren wird in dem Datensatz eine Liste mit den Identifikatoren der Fahrzeuge hinterlegt. Im Unter-

schied zu SOTIS wird als Zeitstempel nicht der Berechnungszeitpunkt, sondern der Empfangszeitpunkt der ältesten verwendeten Basisinformation herangezogen.

CASCADE [78, 77] ist ein weiteres dezentrales Verfahren zum Austausch von Verkehrsinformationen. Wie in den vorherigen Ansätzen auch überträgt jedes Fahrzeug seine eigenen Basisinformationen in regelmäßigen Abständen zu den benachbarten Autos. Mehrere Datensätze von Fahrzeugen eines Straßenabschnitts werden zu einem Cluster-Datensatz kombiniert, welcher den Median der Geschwindigkeit, die durchschnittliche Position und die kompakte Version der verwendeten Basisdatensätze enthält. In diesen kompakten Datensätzen ist sowohl die Position als auch die Geschwindigkeit relativ zu den im Cluster-Datensatz hinterlegten Informationen codiert. Zusätzlich ist in diesen Datensätzen auch der jeweilige Identifikator des Fahrzeugs gespeichert. Wie bei TrafficView dient die Zusammenfassung der Einzeldaten primär der Reduktion der zu übertragenden Datenmenge. Dies ist vor allem daran zu erkennen, dass die im Cluster-Datensatz enthaltenen Basisdaten – abgesehen vom Beobachtungszeitpunkt – verlustfrei wieder extrahiert werden können. In [76] wird der zusätzliche Einsatz eines verlustlosen Kompressionsverfahrens untersucht.

Die Autoren von [164] schlagen ein dezentrales System zur Routenplanung vor. Wie bei SOTIS werden darin die Straßen in Abschnitte unterteilt, wobei allerdings nicht die Geschwindigkeit der Fahrzeuge, sondern die für einen Straßenabschnitt benötigte Reisedauer ausgetauscht wird. Zudem werden in diesem System neuere Beobachtungen nicht dafür verwendet, die alten Daten zu ersetzen, sondern sie werden mit diesen zu einem gleitenden Durchschnitt verrechnet. Hierdurch kann nicht nur die aktuelle Verkehrssituation eingeschätzt werden, sondern auch deren Verbesserung oder Verschlechterung. Des Weiteren können die durchschnittlichen Reisezeiten direkt für die Routenplanung verwendet werden.

Ein Nachfolger des FleetNet-Projekts war *Network on Wheels* (NOW) [51]. Eine Zielsetzung dieses Projekts war unter anderem, die bereits von SOTIS bekannte dezentrale Aggregierung zu verallgemeinern und auch für andere Anwendungszwecke zu nutzen. Hierbei stand allerdings nicht die Frage im Vordergrund, wie genau die Daten aggregiert werden sollen, sondern vielmehr, wie die Interessen mehrerer parallel und unabhängig voneinander arbeitender Anwendungen vereinbart werden können. Da im automobilen Umfeld zahlreiche Anwendungen die gleichen Daten benötigen, sollte es unbedingt vermieden werden, dass jede Anwendung unkoordiniert eigene Nachrichten verschickt und so ein Ressourcenwettbewerb entsteht. Daher wurde vorgeschlagen, dass derartige Anwendungen nur über ein *SLOPE* [136, 137, 59] genanntes Framework Nachrichten austauschen können. Die Anwendungen teilen in diesem Fall SLOPE mit, welche Daten mit welcher Häufigkeit verschickt werden und welche empfangenen Daten an sie weitergeleitet werden sollen. Auf diese Weise werden von mehreren Anwendungen benötigte Informationen wie beispielsweise die Position und die Geschwindigkeit des Fahrzeugs nicht mehrfach übertragen. Außerdem können im Notfall durch diese Architektur Warnungen priorisiert werden. Die eigentliche Aggregierung kann je nach Bedarf zentral in SLOPE oder in angepasster Form direkt in den einzelnen Anwendungen erfolgen.



Alle bisher in diesem Abschnitt vorgestellten Verfahren basieren darauf, lokale Informationen wie beispielsweise die eigene Geschwindigkeit und Position mit den benachbarten Verkehrsteilnehmern auszutauschen. Die Empfänger kombinieren die gesammelten Basisdaten miteinander und übertragen die so entstehenden Aggregate ebenfalls an ihre Nachbarn. Die Aggregierung hat dabei primär den Zweck, die einzelnen Daten zu einer semantisch sinnvollen Beobachtung zu kombinieren und sie in kompakter Form weiter entfernten Fahrzeugen bereitzustellen. Die Autoren von [133] haben festgestellt, dass ein Aggregierungsschema nur dann skaliert, wenn für die benötigte Bandbreite  $o(1/d^2)$  gilt, wobei  $d$  die Distanz zwischen dem Ort der Beobachtung und dem Interessenten bezeichnet. Da in den bisher besprochenen Verfahren nicht vorgesehen ist, die Aggregate wiederum miteinander zu kombinieren, ist das Verhältnis zwischen der Größe des Beobachtungsgebiets und der Größe der Datenstruktur konstant. Somit kann die erforderliche Bandbreitenbeschränkung nur erfolgen, indem die Häufigkeit des Nachrichtenaustauschs mit steigendem Abstand sehr stark abnimmt. Ob derart selten übertragene Informationen den Fahrern allerdings noch nützen können, ist fraglich. Aus diesem Grund beschränken die vorgestellten Verfahren den Verbreitungsbereich auf einige Kilometer und gehen teilweise auch nicht von einer omnidirektionalen Ausbreitung aus, wodurch die zu übertragene Informationsmenge ebenfalls reduziert wird.

Ein weiteres Problem bei TrafficView und CASCADE besteht darin, dass mit steigender Anzahl von Fahrzeugen im Beobachtungsgebiet auch die Anzahl der Basisdaten steigt. So können die Aktualität und Güte der Aggregate durch die in der Datenstruktur hinterlegten Identifikatoren der Fahrzeuge zwar besser abgeschätzt werden, es steigt hierdurch aber auch der benötigte Speicherplatz linear zur Anzahl der enthaltenen Basisdaten. Bei SOTIS können dagegen die Fahrzeuge aufgrund der überschneidungsfreien Gebietsaufteilung und der verwendeten Funkreichweite die gesamte Verkehrssituation auf dem aktuell befahrenen Straßenabschnitt mit ausreichender Genauigkeit einschätzen. Daher ist es ausreichend, beim Vergleich mehrerer Aggregate des gleichen Straßenabschnitts die Güte der einzelnen Aggregate anhand ihres Zeitstempels abzuschätzen. Auf diese Weise kann auf eine Speicherung der Identifikatoren innerhalb der Aggregate verzichtet werden.

Ein Aggregierungsschema kann dementsprechend nur skalieren, wenn die Größe der Aggregate unabhängig von der Anzahl der Fahrzeuge ist und wenn die Aggregate unterschiedlicher Beobachtungsgebiete mehrstufig miteinander aggregiert werden können. In [30] wird ein derartiges Aggregierungsschema zur dezentralen Parkplatzsuche beschrieben. Dieser Ansatz geht grundsätzlich davon aus, dass größere Parkplätze oder Parkhäuser mit einem Zahlungssystem ausgestattet sind und somit die Anzahl der aktuell verfügbaren Parkplätze bekannt ist. Diese Belegungsinformationen werden per Broadcast an alle Fahrzeuge in der Umgebung übertragen. Wie in den bisherigen Ansätzen besitzt auch in diesem Verfahren jedes Fahrzeug einen Zwischenspeicher, in dem empfangene Informationen gesichert und weiterverarbeitet werden können.

Das gesamte Beobachtungsgebiet wird als höchste Aggregierungsebene aufgefasst, die in vier gleich große, sich nicht überschneidende Untergebiete zerlegt wird. Diese werden wie-

derum nach den gleichen Regeln in vier Gebiete aufgeteilt. Dieser Vorgang wird so lange wiederholt, bis die gewünschte Größe erreicht wurde oder jedes Gebiet maximal einen Parkplatz beinhaltet. Dieses Aufteilungsschema kann als Quadtree aufgefasst werden, wobei die Wurzel dem gesamten Beobachtungsgebiet entspricht. Die vier Kinder wiederum entsprechen den Untergebieten des durch den jeweiligen Vater repräsentierten Gebiets. Gemäß dieser Organisationsstruktur werden die Informationen der vier Gebiete mit gleichem Vater miteinander zu einem Aggregat kombiniert, welches zur Bewertung der Parkplatzsituation im nächsthöheren Gebiet herangezogen werden kann.

Die regelmäßig zwischen den Fahrzeugen ausgetauschten Pakete enthalten sowohl eine Liste mit den Basisinformationen der Parkplätze als auch eine Liste mit bereits aggregierten Informationen unterschiedlicher Aggregierungsebenen. Die zu sendenden Basisinformationen werden in Abhängigkeit von der Entfernung des Parkhauses und der Aktualität der Information ausgewählt. Bei den bereits aggregierten Informationen werden vorrangig diejenigen Aggregate ausgewählt, in deren Beobachtungsgebiet sich das Fahrzeug gerade befindet. Die von anderen Fahrzeugen empfangenen Informationen werden – sofern sie aktueller sind – im lokalen Zwischenspeicher hinterlegt. Anschließend werden die aktualisierten Informationen genutzt, um die Aggregate der nächsthöheren Ebene zu erneuern, wobei als Zeitstempel der Durchschnitt der Zeitstempel der tiefer liegenden Ebene verwendet wird.

In den Aggregaten ist neben der Kennzeichnung, auf welche Aggregierungsebene und welches Gebiet sich die Datenstruktur bezieht, auch die Anzahl der insgesamt vorhandenen Parkplätze hinterlegt. Des Weiteren enthält die Datenstruktur Angaben zur aktuellen Auslastung der Parkplätze und einen Zeitstempel. Der benötigte Speicherplatz ist somit sowohl von der Anzahl der Parkhäuser und Fahrzeuge als auch von der Größe des Beobachtungsgebiets unabhängig. Mit steigendem Abstand zum Ursprung der Basisinformation steigt auch die bei der Weiterleitung verwendete Aggregierungsebene, wodurch sich gleichzeitig die zu übertragende Datenmenge pro Fläche reduziert. Bei einer geeigneten Wahl der Konfigurationsparameter kann so die weiter oben genannte Limitierung der Bandbreite auf  $o(1/d^2)$  erreicht werden.

Ein Nachteil des beschriebenen Aggregierungsschemas besteht in dem Umstand, dass die Aktualität und Güte der Aggregate ausschließlich anhand ihrer Zeitstempel eingeschätzt werden können. Da diese den durchschnittlichen Zeitstempeln der tiefer liegenden Aggregierungsebenen entsprechen, hängen diese auch direkt von der Aktualität der jeweiligen Basisinformationen ab. Nimmt man an, dass ein Fahrzeug ein Aggregat aus wenigen aktuellen Basisinformationen berechnet hat, so ist dessen durchschnittlicher Zeitstempel ebenfalls aktuell. Zeitnah könnte ein anderes Fahrzeug die gleichen Basisinformationen mit zusätzlichen Beobachtungen anderer Parkhäuser kombiniert haben, die allerdings etwas älter sind. In diesem Fall beinhaltet das zweite Aggregat mehr Informationen als das erste, weist aber zugleich einen älteren Zeitstempel auf. Selbst unter der Voraussetzung, dass der Empfänger beider Aggregate die Anzahl der enthaltenen Basisinformationen kennt, kann nur vage abgeschätzt werden, welches Aggregat die höhere Güte besitzt.



Ein weiterer Nachteil dieses Ansatzes liegt in dem Umstand, dass Aggregate, nachdem sie aus den Aggregaten der tiefer liegenden Aggregierungsebene berechnet wurden, nicht mehr aktualisiert werden können. Es ist beispielsweise nicht möglich, zwei gleichwertige Aggregate zu einem Aggregat der gleichen Ebene zu vereinigen. Ebenfalls können vorliegende Basisinformationen nicht im Nachhinein in ein empfangenes Aggregat eingefügt werden. Aufgrund dieses Sachverhalts werden von den Fahrzeugen immer Aggregate aller Aggregierungsebenen ausgetauscht, wodurch allerdings auch redundante Informationen übertragen werden. Um diese Limitierungen zumindest teilweise zu überwinden, wird in [105] vorgeschlagen, die Beobachtungen in einer probabilistischen Datenstruktur zu hinterlegen. Dabei wird beispielsweise die Summe aller freien Parkplätze innerhalb des jeweiligen Beobachtungsgebiets nicht mehr als Zahl gespeichert, sondern in Form von *Flajolet-Martin-Sketches* [55]. Aufgrund der Eigenschaften dieser Datenstruktur können diese Aggregate sowohl mit Aggregaten der gleichen Aggregierungsebene als auch mit Basisinformationen verlustfrei kombiniert werden, wodurch jegliche Abwägung der Aktualität oder Güte eines Aggregats entfällt. Allerdings besitzt die probabilistische Repräsentation eine inhärente Ungenauigkeit. Je mehr Speicherplatz für die Datenstruktur verwendet wird, umso genauer kann die Summe der Basisbeobachtungen bestimmt werden. In [105] verwenden die Autoren Aggregate der Größe von etwa zwanzig Byte und erreichen hierbei eine zufriedenstellende Genauigkeit. Hierbei ist zu berücksichtigen, dass die anderen Aggregierungsschemata ebenfalls inhärente Ungenauigkeiten aufweisen. Beispielsweise führen die Berechnung des Zeitstempels für die nächsthöhere Ebene oder die Auswahlstrategie für das Aggregat, welches an die Nachbarn übertragen werden soll, immer zu Abweichungen.

## 2.3 Zusammenfassung

In diesem Kapitel wurden die beiden grundsätzlichen Kommunikationsansätze für drahtlose Ad-hoc-Netzwerke beschrieben. Die erste Strategie besteht darin, gezielt Pfade zwischen zwei nicht benachbarten Kommunikationspartnern zu finden und alle dazwischen liegenden Knoten ausschließlich als Weiterleiter zu verwenden. In Kapitel 3 wird mit GRANT ein neues derartiges Protokoll vorgestellt und mit einigen der zuvor beschriebenen Verfahren in einer großen Simulationsstudie verglichen.

Der zweite Kommunikationsansatz verzichtet auf die explizite Adressierung von Knoten. Stattdessen werden die Informationen an alle Nachbarn übertragen, die dann gemäß eines verteilten Algorithmus entscheiden, ob und wie sie diese Information weiterverarbeiten und weiterleiten. In diesem Kapitel wurden einige derartige Anwendungen exemplarisch beschrieben. Sollen die Daten auch großflächig verbreitet und kombiniert werden, wird für die Datenreduktion ein geeignetes Aggregierungsschema benötigt. Ein Lösungsansatz hierfür ist die Verwendung von probabilistischen Datenstrukturen, die nur gewisse statistische Merkmale der Informationen speichern, sodass der benötigte Speicherplatz signifikant sinkt. In Kapitel 4 wer-

den deshalb zwei neue probabilistische Datenstrukturen vorgeschlagen und mit bestehenden Ansätzen verglichen.

## Kapitel 3

# Positionsbasierte Routingverfahren für den Einsatz in Innenstädten

Ausdauer wird früher oder später  
belohnt – meistens aber später.

---

– Wilhelm Busch –

Die in aktuellen Netzwerken überwiegend verwendete Kommunikationsform sieht den Datenaustausch zwischen exakt zwei Kommunikationspartnern vor. Hierfür werden die jeweiligen Nutzdaten in kleine Datenpakete zerlegt und zu dem anderen Kommunikationspartner übertragen. Jedes gesendete Paket kann von allen anderen Geräten der gleichen Kollisionsdomäne empfangen werden. Ist ein Gerät beispielsweise an ein Netzkabel angeschlossen und sendet über dieses ein Paket, so bilden alle anderen ebenfalls direkt angeschlossenen Geräte eine Kollisionsdomäne und können das übertragene Paket automatisch mithören. Bei drahtlosen Übertragungstechniken wird die Luft als Übertragungsmedium verwendet. Dabei befinden sich dann alle anderen Geräte im Umkreis der Funkreichweite in der gleichen Kollisionsdomäne wie der Sender und können ein übertragenes Paket mithören. Soll ein Paket zu einem Gerät übertragen werden, welches sich nicht in der gleichen Kollisionsdomäne befindet, so muss das Paket von anderen an der Kommunikation unbeteiligten Geräten zum Ziel weitergeleitet werden. Dieser schrittweise Weiterleitungsprozess wird Routing genannt.

In drahtgebunden Netzwerken werden die Pakete typischerweise anhand der Topologie weitergeleitet. Wie in Kapitel 2.1 dargestellt, kann topologiebasiertes Routing auch für mobile Ad-hoc-Netzwerke eingesetzt werden. Da viele mobile Geräte mit einem Positionierungssystem wie zum Beispiel GPS ausgestattet sind, kann beim Routing alternativ auch die Position der Geräte verwendet werden. Dieser Ansatz hat in zahlreichen Publikationen vielversprechende Ergebnisse erzielt, allerdings meist in vergleichsweise kleinen Simulationsgebieten. Außerdem wird in fast allen Veröffentlichungen zu wenig Augenmerk darauf gelegt, dass die meisten positionsbasierten Verfahren nur Heuristiken sind und selbst für durchgängig miteinander verbundene Kommunikationspartner keine erfolgreiche Pfadsuche garantieren können. Da aber üblicherweise in den Simulationen alle nicht zugestellten Pakete als Verluste gewertet werden, kann nicht mehr ohne Weiteres differenziert werden, ob die Pakete aufgrund eines Netzwerkproblems wie beispielsweise einer Überlastung oder einer zeitweiligen Fragmentierung oder aber aufgrund der Heuristik selbst nicht zugestellt werden konnten. Daher wird in diesem Ka-

pitel ein neues Simulationsmodell vorgeschlagen, welches speziell für die großflächige Analyse derartiger Routingheuristiken optimiert ist. Da die im Weiteren vorgestellten Ergebnisse einen großen Verbesserungsbedarf aufzeigen, wird zusätzlich ein positionsbasiertes Routingverfahren vorgestellt und mit den bestehenden Ansätzen verglichen.

### 3.1 Verallgemeinerung bestehender positionsbasierter Routingverfahren

Wird in einem Ad-hoc-Netzwerk ein Knoten eingeschaltet, hat dieser kein Wissen über die Topologie des Netzwerks. Der erste Schritt besteht deshalb meist darin, die lokale Nachbarschaft durch den Austausch sogenannter Beacon-Pakete zu erkunden. Jeder Knoten befindet sich dann mit den Empfängern seiner Beacon-Pakete in einer räumlich begrenzten Kollisionsdomäne. Um Knoten zu finden, die sich nicht in der gleichen Kollisionsdomäne befinden, verwenden topologiebasierte Routingverfahren das sogenannte Fluten. Ein Flutprozess wird gestartet, indem ein Knoten ein entsprechendes Paket per Broadcast an seine direkten Nachbarn schickt. Ist der Empfänger nicht der gesuchte Knoten und hat er zuvor noch keine Kopie des Pakets erhalten, so wiederholt er die lokale Suche, indem er das Paket wiederum an seine direkten Nachbarn weiterleitet. Dieser verteilte Prozess endet meist erst dann, wenn alle Knoten des Netzwerks eine Kopie des Pakets erhalten haben. Um die Belastung des Netzwerks zu reduzieren, wird oft der Suchradius limitiert. Jeder Empfänger erhöht in einem solchen Fall den im Paketkopf mitgeführten Hop-Zähler um eins und leitet das Paket nur weiter, wenn der Hop-Zähler kleiner als der gewählte Schwellenwert ist. Wurde der gesuchte Knoten gefunden, so ist auch der zurückgelegte Pfad zwischen dem Initiator der Suche und dem Zielknoten bekannt. Topologiebasierte Verfahren leiten die zu transportierenden Datenpakete entlang der durch Fluten gefundenen Pfade weiter. Die Pfade können hierbei entweder vom Sender vollständig in den Paketkopf eingetragen werden (DSR) oder sie liegen ausschließlich in verteilter Form in den lokalen Routingtabellen der Knoten vor (AODV). Sollte ein Paket aufgrund topologischer Veränderungen, zum Beispiel durch Knotenbewegungen, nicht mehr entlang des hinterlegten Pfades weitergeleitet werden können, wird mit Hilfe einer räumlich begrenzten Flutoperation ein neuer Pfad gesucht.

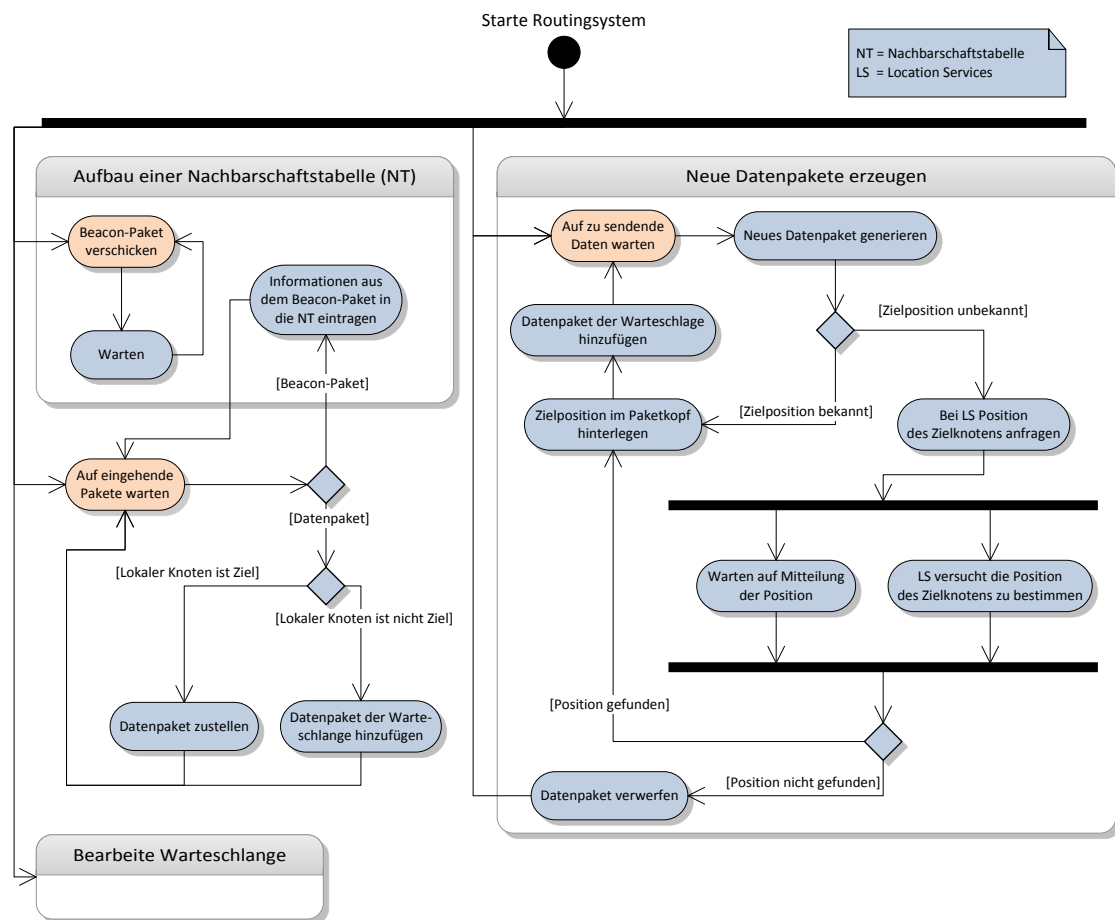
Auch positionsbasierte Routingverfahren müssen zur Suche nach unbekannten Knoten beziehungsweise deren Positionen auf das Fluten zurückgreifen. Der grundlegende Unterschied zu topologiebasierten Verfahren besteht aber darin, dass während der Suche keine explizite Route zum Zielknoten aufgebaut wird. Stattdessen wird die aktuelle Position des gefundenen Knotens an den Initiator der Suche weitergeleitet. Jeder Weiterleiter entscheidet dann anhand der Position des Zielknotens, zu welchen direkten Nachbarn das eigentliche Datenpaket im Folgenden weitergeleitet werden soll, um den Zielknoten zu erreichen. Abgesehen von der Zielposition wird bei diesem Auswahlprozess auch zusätzliches Wissen wie beispielsweise über die Positionen der direkten Nachbarn verwendet. Dahinter steht die Idee, dass schon die Bewegung eines einzelnen Knotens die expliziten Routen veralten lassen kann, wohingegen

die Position des Zielknotens nur durch seine eigene Bewegung beeinflusst wird und somit im Durchschnitt langsamer altert als ein expliziter Pfad. Gerade in hoch mobilen Umgebungen wie zum Beispiel auf Autobahnen ergeben sich hierdurch Vorteile [57, 58]. Allerdings sind viele positionsbasierte Routingverfahren nur Heuristiken, die versuchen, mit lokalem Wissen den am besten geeigneten Weiterleiter auszuwählen, ohne dabei garantieren zu können, dass sie immer einen Pfad zum Zielknoten finden. Um diese Limitierung zu überwinden, kann eine effiziente positionsbasierte Heuristik durch ein zweites Verfahren ergänzt werden, welches die erfolgreiche Pfadsuche garantieren kann. Basierend auf den vielen positionsbasierten Ansätzen ergeben sich zahlreiche Kombinationsmöglichkeiten, die in diesem Kapitel genauer untersucht werden. Der erste Schritt hierfür liegt in der Verallgemeinerung bestehender positionsbasierter Routingverfahren.

Der verallgemeinerte Ablauf von positionsbasiertem Routing wird in Abbildung 3.1 aus der Sicht eines einzelnen Knotens in Form eines UML-Aktivitätsdiagramms visualisiert. Der dargestellte Ablauf orientiert sich hierbei an Verfahren, die proaktiv Beacon-Pakete austauschen, um eine Nachbarschaftstabelle aufzubauen. Allerdings sind die meisten Schritte auch für die in Unterabschnitt 2.1.2 besprochenen wettbewerbsorientierten Verfahren anwendbar, die auf den Austausch von Beacon-Paketen verzichten. Insgesamt besteht der dargestellte Ablauf aus vier nebenläufigen Vorgängen, von denen einer in Abbildung 3.2 ausgelagert wurde. Die jeweils erste Aktion von den vier Vorgängen wurde in den Diagrammen farblich hervorgehoben.

Einer der nebenläufigen Vorgänge besteht darin, in regelmäßigen Zeitabständen Beacon-Pakete mit der eigenen Position und dem Identifikator des Knotens an alle Nachbarn zu übertragen. Empfangene Beacon-Pakete werden von den Knoten verwendet, um eine lokale Nachbarschaftstabelle aufzubauen und zu pflegen. Ist ein eingehendes Datenpaket für den lokalen Knoten bestimmt, so wird es an die entsprechende Anwendung weitergereicht. Anderenfalls wird es in die lokale Warteschlange mit den zu bearbeitenden Paketen eingefügt. Daten von lokalen Anwendungen werden in neue Datenpakete verpackt und ebenfalls dieser Warteschlange hinzugefügt. Dafür muss aber die Position des Ziels im Paketkopf hinterlegt worden sein. Ist diese Position nicht a priori bekannt, muss sie über einen Flutmechanismus ermittelt werden. Häufig wird diese Funktionalität in eine eigenständige Softwareinstanz ausgelagert, die *Location Service* genannt wird.

Das Bearbeiten der Warteschlange mit den zu sendenden Datenpaketen ist der umfangreichste Vorgang und wird in Abbildung 3.2 dargestellt. Anhand des aus der Warteschlange entnommenen Pakets und der lokalen Informationen wie beispielsweise der Nachbarschaftstabelle wird nun entschieden, an welchen Knoten das Paket im nächsten Schritt weitergeleitet wird. Manche Verfahren setzen für ihre Funktionsweise einen planarisierten Netzwerkgraphen voraus. Die zu diesem Zweck verwendeten Teilgraphen können mit lokalem Wissen bestimmt werden, indem alle direkten Nachbarn ignoriert werden, die nicht den entsprechenden Kriterien des Teilgraphen entsprechen. Aus diesem Grund kann das Planarisierungsverfahren als eine auf die Nachbarschaftstabelle angewendete Filteroperation angesehen werden. Die Eignung der verbleibenden Nachbarn wird anhand einer Bewertungsfunktion beurteilt. Im Fall



**Abbildung 3.1:** Schematische Darstellung von positionsbasiertem Routing in Form eines UML-Aktivitätsdiagramms.

des *Greedy Routing Scheme* (GRS) wird hierfür beispielsweise der geometrische Abstand zwischen dem Nachbarn und dem Zielknoten herangezogen. Der Knoten mit der besten Bewertung wird anschließend als Weiterleiter ausgewählt. Um Routingschleifen zu vermeiden, überprüfen manche Verfahren vor dem Weiterleiten eine Abbruchbedingung. Üblicherweise wird dabei die Eignung des aktuellen Knotens mit der des ausgewählten Nachbarn verglichen. Sollte der aktuelle Knoten eine bessere Bewertung erzielen, so befindet sich der Weiterleitungsprozess in einem sogenannten lokalen Optimum und wird unterbrochen. Zur Überbrückung des lokalen Optimums kann eine alternative Weiterleitungsstrategie eingesetzt werden. Hierbei wird das Paket so lange weitergeleitet, bis ein Knoten erreicht wurde, der eine bessere Bewertung erzielt als der Knoten, der den Überbrückungsprozess gestartet hat. Diese Schritte werden in der Abbildung unter dem Punkt „Überprüfe Moduswechsel“ zusammengefasst. Als Überbrückungsstrategien können beispielsweise der Face-2-Algorithmus oder ein topologiebasiertes Verfahren eingesetzt werden.

### 3.1 Verallgemeinerung bestehender positionsbasierter Routingverfahren

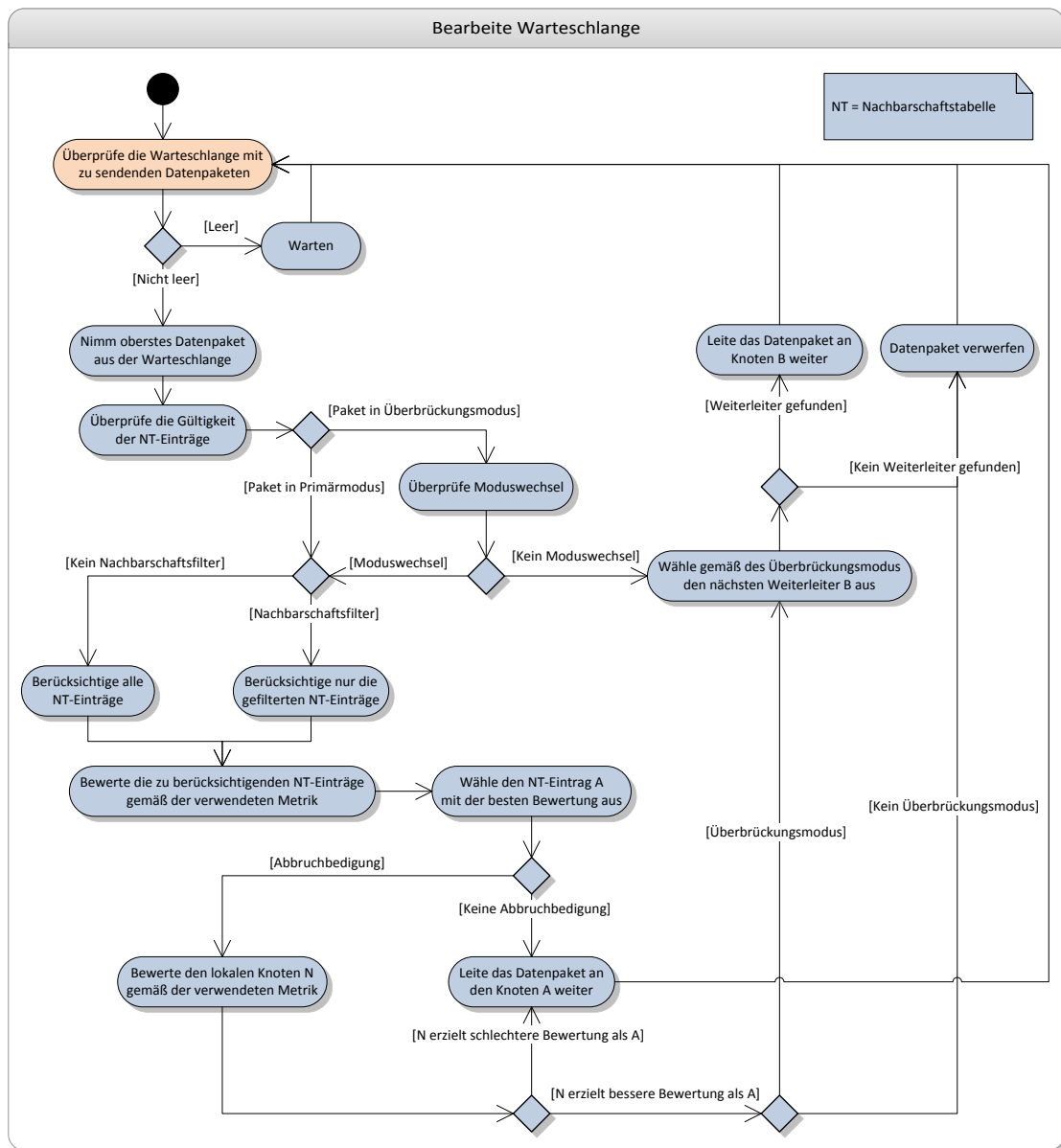


Abbildung 3.2: Bearbeitung der zu sendenden Datenpakete.

**Tabelle 3.1:** Etablierte positionsbasierte Routingverfahren als Kombinationen aus Bewertungsfunktion, Nachbarschaftsfilter und Überbrückungsfunktion.

| Name     | Bewertung                      | NT-Filter | Überbrückung                           | Zustellung  |
|----------|--------------------------------|-----------|--|-------------|
| GRS      | Abstand                        | -         | -                                      | Heuristisch |
| MFR      | Abstand                        | -         | -                                      | Heuristisch |
| NFP      | Abstand                        | -         | -                                      | Heuristisch |
| Compass  | Winkel                         | GG/RNG/-  | -                                      | Heuristisch |
| Face-2   | Winkel                         | GG/RNG    | -                                      | Garantiert  |
| RGR      | Abstand mit Kreuzungserkennung | -         | Face-2 auf Basis der Straßenkreuzungen | Garantiert  |
| GFG/GPSR | Abstand                        | -         | Face-2                                 | Garantiert  |
| PBR-DVR  | Abstand                        | -         | Topologiebasiert                       | Garantiert  |

Der vorgestellte Ablauf kann genutzt werden, um nahezu alle in Abschnitt 2.1 präsentierten Routingverfahren abzubilden. Tabelle 3.1 zeigt für jedes Verfahren die jeweilige Kombination aus Bewertungsfunktion, Nachbarschaftsfilter und Überbrückungsfunktion. Im Unterschied zu allen anderen Verfahren wurde *Restricted Greedy Routing* (RGR) speziell für den Einsatz in Innenstädten konzipiert und basiert auf einer Straßenkreuzungserkennung, die sowohl im Primärmodus als auch in der Überbrückungsstrategie herangezogen wird. Im Primärmodus werden Knoten auf entdeckten Straßenkreuzungen bevorzugt ausgewählt, selbst wenn sich in der Funkreichweite ein dem Ziel näher liegender Knoten befinden sollte. Diese Strategie baut auf dem Grundgedanken auf, dass Knoten auf Kreuzungen anhand ihrer Position entscheiden können, entlang welcher angrenzenden Straße ein Paket weitergeleitet werden soll. Die dafür notwendige Kreuzungserkennung basiert auf einer Analyse der lokalen Nachbarschaftstabellen. Eine Kreuzung kann somit nur erkannt werden, wenn sich auch Knoten in den angrenzenden Straßen befinden. Auch wenn dieser Mechanismus für Innenstadtszenarien gedacht ist, kann er doch in jedem Ad-hoc-Netzwerk verwendet werden. Eine Kreuzung kann dann als Gabelungspunkt interpretiert werden, von dem aus ein Paket in unterschiedliche Richtungen weitergeleitet werden kann. Da diese Gabelungspunkte generell von hoher Bedeutung sind, könnte es sinnvoll sein, diese Strategie ebenfalls mit anderen heuristischen Ansätzen wie beispielsweise *Most Forward within Radius* (MFR) oder Compass zu kombinieren. Des Weiteren kann jede dieser Kombinationen mit einer Überbrückungsstrategie ergänzt werden. Die Eignung der sich daraus ergebenden Protokollkombinationen wird im Folgenden analysiert und evaluiert.

### 3.2 Erstellen einer geeigneten Simulationsumgebung

Zur Analyse von Routingprotokollen werden häufig Netzwerksimulatoren herangezogen. Prinzipiell ist es dabei wünschenswert, alle Einflussfaktoren so exakt wie möglich in der Simulationsumgebung nachzubilden. Trotz der hohen Rechenleistung aktueller Prozessoren ist dies



allerdings nicht in vollem Umfang möglich. Neben der Berechnungsdauer besteht ein weiteres Problem darin, dass nicht alle Einflussfaktoren bekannt beziehungsweise modellierbar sind. So kann die Signalausbreitung von drahtlosen Übertragungen nur berechnet werden, wenn auch alle Umgebungsfaktoren wie Bebauung, verwendete Materialien, Strahlenquellen, Witterungsbedingungen und so weiter mit hinreichender Genauigkeit bekannt sind. Des Weiteren gibt es gerätespezifische Einflussfaktoren, die ohne Zugriff auf die Software und Hardware eines Gerätes nicht exakt nachgebildet werden können. Beispielsweise unterscheiden sich die Sende- und Empfangscharakteristiken unterschiedlicher WLAN-Geräte voneinander. Aus diesen Gründen ist es im Normalfall notwendig, die Komplexität der realen Zusammenhänge durch die Nutzung von hinreichend genauen Modellen zu reduzieren.

In den folgenden Unterabschnitten werden die wesentlichen Bestandteile einer Simulationsumgebung für mobile Ad-hoc-Netzwerke besprochen.

### Netzwerksimulator

Ein Netzwerksimulator dient dazu, ein gesamtes Netzwerk mit all seinen Bestandteilen zu simulieren. Dies umfasst sowohl die Hardwarekomponenten wie beispielsweise das Funkmodul als auch die Softwarebestandteile. Zu Letzteren zählen insbesondere die Implementierungen der einzelnen Protokolle auf allen relevanten Netzwerkschichten. Für die Simulation eines Routingprotokolls sind die Implementierungen der physikalischen Schicht und der Sicherungsschicht des IEEE-Standards 802.11 [80] von besonderem Interesse.

Die meisten Simulatoren verwenden ein ereignisorientiertes Simulationsmodell, das aus einem Zustandsmodell, einer Ereignisliste und Ereignisroutinen besteht. Während des Simulationsverlaufs werden die in der Ereignisliste enthaltenen Ereignisse durch die Ausführung der jeweiligen Ereignisroutinen chronologisch abgearbeitet. Dabei können die eingetretenen Ereignisse wiederum neue Vorkommnisse in der Zukunft auslösen.

Bekannte freie Netzwerksimulatoren sind unter anderem *ns-2* [11], *ns-3* [12], *SWANS* [5], *GloMoSim* [7] und *Shawn* [49, 16]. Des Weiteren gibt es auch zahlreiche kommerzielle Produkte wie beispielsweise *QualNet* [15] und *OMNeT++* [13] (kostenlos für den akademischen Einsatz).

### Funkwellenausbreitungsmodell

Das Funkwellenausbreitungsmodell wird vom Netzwerksimulator herangezogen, um zu entscheiden, von welchen Geräten ein abgeschicktes Paket erfolgreich empfangen werden kann. Abhängig von Eingabeparametern, beispielsweise dem Abstand zwischen den Kommunikationspartnern, der Höhe der Antennen oder der Sendeleistung, ermitteln deterministische Modelle, mit welcher Empfangsleistung die gesendeten Daten ankommen. Liegt die Empfangsleistung über einem festzulegenden Schwellenwert, so gelten die Daten als erfolgreich decodiert und werden an die höheren Netzwerkschichten weitergeleitet. Welcher Schwellenwert genutzt werden soll, hängt unter anderem von der zu simulierenden Hardware und der verwendeten

Modulationstechnik ab. Die Berechnung bei deterministischen Modellen wie beispielsweise dem *Free Space Model* oder dem *Two Ray Ground Model* [127] erfolgt ohne Verwendung einer Zufallsverteilung. Dementsprechend berechnen diese Modelle bei gleichbleibenden Eingabeparametern immer dieselbe Empfangsleistung, die ab einem gewissen Abstand vom Sender unter den festgelegten Schwellenwert fällt. Abgesehen von Paketkollisionen und anderen Störungen kann somit ein Netzwerknoten mit allen anderen Knoten innerhalb des maximalen Abstands – welcher folglich der Übertragungsreichweite entspricht – fehlerfrei kommunizieren. Der sich aus diesem Sachverhalt ergebende Netzwerkgraph entspricht dem in Abschnitt 2.1 vorgestellten *Unit Disk Graph* (UDG).

Neben den deterministischen Modellen gibt es auch auf Wahrscheinlichkeitsverteilungen basierende Ansätze. Bekannte probabilistische Verfahren sind unter anderem das *Nakagami Model* [111] und das *Log-Normal Shadowing Model* [127]. Gemäß dem probabilistischen Ansatz kann ein gesendetes Paket nur mit einer gewissen Wahrscheinlichkeit von benachbarten Knoten empfangen werden. Diese Wahrscheinlichkeit sinkt mit steigendem Abstand der Kommunikationspartner zueinander langsam auf null. Je nachdem, welche Parameter für die zugrunde liegende Wahrscheinlichkeitsverteilung gewählt wurden, können auch unmittelbar benachbarte Netzteilnehmer nur mit einer Wahrscheinlichkeit von deutlich unter 100 % Pakete austauschen. Somit gibt es bei probabilistischen Modellen keine klassische Übertragungsreichweite.

Der reale drahtlose Datenaustausch verhält sich nicht rein deterministisch und wird sehr stark von äußeren Faktoren beeinflusst. Allerdings sind die am meisten genutzten probabilistischen Ansätze letztlich nur Versuche, reale Messergebnisse durch ein mathematisches Modell abzubilden. Die dabei gewählten Parameter sind aber nicht für jede Umgebung identisch, sodass sich ein hoher Konfigurationsspielraum ergibt. Eine Übersicht über unterschiedliche Funkwellenausbreitungsmodelle und deren Einfluss auf die Empfangswahrscheinlichkeiten kann [151] oder [97] entnommen werden.

### Bewegungsmodell

Für die Simulation eines drahtgebunden Netzwerks wird dem Simulator die Topologie vorgegeben. Hierdurch ist dem Netzwerksimulator bekannt, wie viele Knoten simuliert werden sollen und wie diese untereinander verbunden sind. Da die Knoten bei drahtlosen Netzwerken ihre Positionen verändern können, kann nicht davon ausgegangen werden, dass die Netzwerktopologie während der gesamten Simulationsdauer unverändert bleibt. Aus diesem Grund werden dem Simulator ständig die Positionen der Knoten mitgeteilt; diese nutzt er, um die Kommunikationsmöglichkeiten zwischen den Knoten mit Hilfe des Funkwellenausbreitungsmodells zu bestimmen. Das Bewegungsmodell umfasst deshalb sowohl die Anfangspositionen der Knoten als auch deren Bewegungen während der Simulation.

Im Wesentlichen gibt es drei unterschiedliche Arten von Bewegungsmodellen. Zu der ersten Gruppe zählen alle zufallsbasierten Modelle wie das häufig verwendete *Random Way-point* (RWP) [83]. Gemäß dem RWP-Modell werden zu Beginn der Simulation die Knoten

gleichmäßig über die hindernisfreie Simulationsebene verteilt. Anschließend wählt jeder Knoten eine zufällige Zielposition innerhalb des Simulationsgebiets aus und bewegt sich direkt auf diese zu. Die dabei verwendete Geschwindigkeit wird gleich verteilt aus dem gewünschten Geschwindigkeitsintervall ausgewählt. Sobald ein Knoten an seiner Zielposition angekommen ist, wählt er nach einer zufälligen Wartezeit das nächste Ziel aus, und der Prozess beginnt von vorne. Die anfangs gewählten statistischen Eigenschaften dieses vergleichsweise einfachen Bewegungsmodells sind allerdings nicht über die gesamte Simulationsdauer konstant. Aufgrund ihrer ursprünglichen Verteilung und des begrenzten Simulationsgebiets wählen die Knoten mit hoher Wahrscheinlichkeit Zielpositionen aus, deren direkter Besuch über die Mitte des Simulationsgebiets führt [128]. Aus diesem Grund sind die Knoten schon nach kurzer Dauer nicht mehr gleichmäßig über das Simulationsgebiet verteilt. Des Weiteren schwankt auch die durchschnittliche Geschwindigkeit der Knoten während des Simulationsverlaufs beträchtlich [166, 115, 22]. Aufgrund dieser Eigenschaften wird in neueren Publikationen, die ein zufallsbasiertes Bewegungsmodell verwenden, häufig auf *Modified Random Direction* (MRD) [128] zurückgegriffen. Bei MRD wird anstatt einer Zielposition ein zufälliger Winkel ausgewählt, in dem sich der Knoten von seiner aktuellen Position aus fortbewegt. Sowohl die Dauer dieser Bewegung als auch die Geschwindigkeit werden ebenfalls zufällig bestimmt. Eine Übersicht über weitere zufallsbasierter Bewegungsmodelle kann [31] entnommen werden.

Eine weitere Möglichkeit der Bewegungsmodellierung besteht darin, reale Bewegungen zu beobachten und aufzuzeichnen. Mit Hilfe dieser Aufzeichnungen können dann die Bewegungsabläufe in der Simulation realistisch nachgebildet werden. Allerdings ist der hiermit verbundene Aufwand sehr groß, insbesondere wenn größere Netzwerke simuliert werden sollen. Des Weiteren sind diese Modelle nicht frei parametrisierbar und können daher nur begrenzt verwendet werden, um unterschiedliche Situationen abzubilden.

Die dritte Möglichkeit besteht in dem Einsatz spezieller Bewegungssimulatoren. Im Unterschied zu den zufallsbasierten Bewegungsmodellen berücksichtigen diese auch Nebenbedingungen wie beispielsweise das Verhalten der anderen Knoten. So hat etwa ein Netzwerkteilnehmer eine gewisse Masse und kann nicht verzögerungsfrei auf eine Wunschgeschwindigkeit beschleunigen. Soll die Bewegung von Fahrzeugen realistisch modelliert werden, so müssen auch die Verkehrsregeln (zum Beispiel die Rechts-vor-links-Regel) und Verkehrszeichen (zum Beispiel Tempolimits oder Ampeln) berücksichtigt werden. Deshalb ist es plausibel, die im Bereich der Verkehrsplanung genutzten Verkehrssimulatoren wie beispielsweise *SUMO* [91] oder *VISSIM* [17] zu verwenden, um realitätsnahe Bewegungsdaten zu generieren. Die einfacheren Hilfsprogramme werten hierfür die von den Verkehrssimulatoren angelegten Protokoll-dateien aus und übertragen die Positionsdaten in ein für den jeweiligen Netzwerksimulator lesbares Format [86, 123, 53]. Die weiter fortgeschrittenen Ansätze modifizieren beide Simulatoren und lassen sie während der Simulation in beide Richtungen Informationen austauschen [157, 102, 146]. Im Unterschied zu den vorab berechneten Bewegungsdaten können die Netzwerkanwendungen bei einer Kopplung der Simulatoren während der Simulation Ein-

fluss auf die Knotenbewegung nehmen. So ist es beispielsweise möglich, in einer verteilten Anwendung einen Stau zu erkennen und nachfolgende Autos umzuleiten.

Anstatt ein theoretisches Bewegungsmodells zu verwenden, kann ein solches Modell auch aus gemessenen Bewegungsdaten extrahiert werden. In [89] und [155] wurden beispielsweise die Bewegungen und Aktivitäten von WLAN-Geräten auf einem Universitätsgelände ausgewertet, um anhand dieser Daten ein Bewegungsmodell zu generieren. Ein anderes Beispiel ist das Modell *META* [74], das auf den GPS-Daten von Taxis aufbaut.

Eine Übersicht über die unterschiedlichen Arten von Bewegungsmodellen ist unter anderem in [19] zu finden. Eine detaillierte Beschreibung und Bewertung aktueller Bewegungsmodelle speziell für die Simulation von Fahrzeugnetzwerken kann [68] entnommen werden.

### Kommunikationsmodell

Das Kommunikationsmodell beschreibt, welche Knoten wann und auf welche Weise miteinander kommunizieren. Dementsprechend gibt dieses Modell vor, zwischen welchen Knoten das Routingprotokoll Pfade finden soll. Abhängig von der Netzwerkdichte und anderen Parametern kann das simulierte Netzwerk allerdings partitioniert sein. Da ein Pfad nur gefunden werden kann, wenn sich beide Knoten in der gleichen Netzwerkpartition befinden, muss dieser Sachverhalt bei der Bewertung der Simulationsergebnisse überprüft werden. Anderenfalls würde die Bewertung des Routingprotokolls von der meist zufällig gewählten Netzwerktopologie beeinflusst und das Ergebnis somit verfälscht werden.

Neben dem Start- und dem Endzeitpunkt ist auch der eigentliche Datenaustausch von hoher Bedeutung. Wenn das Kommunikationsverhalten der höheren Schichten nicht bekannt ist, werden die Daten meist mit einer konstanten Bitrate übertragen. Falls das Routingprotokoll nicht speziell in einer Überlastungssituation analysiert werden soll, muss die Bitrate der Übertragung (Größe und Sendefrequenz der Pakete) so gewählt werden, dass es an keinem Knoten zu einer vollständigen Auslastung der verfügbaren Bandbreite kommt.

### Bewertungskriterien

In Abhängigkeit von den Anforderungen und jeweiligen Bewertungskriterien müssen die verschiedenen Komponenten einer Simulationsumgebung sinnvoll kombiniert werden. Hierbei muss zum einen die Verfügbarkeit berücksichtigt werden, das heißt, es muss geprüft werden, ob beispielsweise das bevorzugte Funkwellenausbreitungsmodell im ausgewählten Simulator vorhanden ist. Zum anderen muss auch die Berechnungskomplexität der Simulation berücksichtigt werden. Typische Bewertungskriterien von Routingprotokollen sind:

- **Die Paketzustellrate:** Sie gibt an, welchen Prozentsatz der abgeschickten Pakete das Routingprotokoll zustellen kann. Hierbei werden nur die Datenpakete der Ende-zu-Ende-Kommunikationen berücksichtigt.

- **Die Verzögerung der Paketzustellung:** Dieses Bewertungskriterium gibt an, wie lange das Weiterleiten vom Startknoten zum Zielknoten im Durchschnitt benötigt. Hierbei kann zwischen der Verzögerung des jeweils ersten Pakets in einem Strom und derjenigen der folgenden Pakete differenziert werden.
- **Die verwendete Bandbreite:** Sie umfasst den gesamten Datenaustausch, der notwendig ist, um den Pfad zum Zielknoten zu finden und die Daten zu diesem weiterzuleiten. Anstelle der gesamten Datenmenge kann auch die Anzahl der gesendeten 1-Hop-Unicast-Pakete herangezogen werden.
- **Die Pfadlänge:** Die Güte der gefundenen Pfade kann anhand der Pfadlänge bewertet werden. Hierfür können sowohl die zurückgelegte Strecke als auch die Anzahl der Weiterleiter (Hops) herangezogen werden. Als Vergleichswert kann beispielsweise der jeweils kürzeste Pfad dienen.

Da die Hauptaufgabe eines Routingprotokolls im zuverlässigen Weiterleiten von Paketen besteht, ist die Paketzustellrate das wichtigste Bewertungskriterium. In diesem Zusammenhang ist es wichtig, im Detail zu analysieren, warum Pakete nicht zugestellt werden konnten. Die wichtigsten Ursachen werden im Folgenden skizziert.

- **Fehlerhafte Protokollzustände:** Die hier betrachteten Routingprotokolle sind alle dezentral organisiert und speichern daher Zustandsinformationen auf den einzelnen Knoten. Die Knotenbewegung verursacht allerdings topologische Veränderungen, die zu ungültigen Protokollzuständen führen können. Bewegt sich beispielsweise ein benachbarter Knoten aus der Funkreichweite heraus, so wird der entsprechende Eintrag in der lokalen Nachbarschaftstabelle ungültig. Durch diesen Umstand werden die über diesen ehemaligen Nachbarn führenden Routen ebenfalls unbrauchbar.
- **Unentdeckte Übertragungsfehler:** Der Empfang von Paketen kann durch Kollisionen oder andere externe Störungen beeinträchtigt werden. Ebenso führt der Einsatz von probabilistischen Funkwellenausbreitungsmodellen zu Paketverlusten. Zwar werden unbestätigte Pakete auf der Datensicherungsschicht erneut übertragen, dennoch kann auch dieser Mechanismus nicht immer eine erfolgreiche Zustellung garantieren. Dementsprechend muss ein Routingprotokoll für den fehlerfreien Betrieb auf Meldungen der tiefer liegenden Schichten reagieren oder den Paketverlust anderweitig feststellen.
- **Keine Konnektivität:** Wie oben bereits erwähnt, können zwei Knoten nur dann miteinander kommunizieren, wenn sie sich in der gleichen Netzwerkpartition befinden. Dementsprechend sollten bei der Berechnung der Paketzustellrate nur diejenigen Datenpakete berücksichtigt werden, die aufgrund der aktuellen Topologie auch zugestellt werden können. Hierfür müssen die Daten des Bewegungs- und des Kommunikationsmodells miteinander kombiniert werden.

Wenn die Erreichbarkeit des Zielknotens sichergestellt ist, können die Paketverluste im Wesentlichen auf fehlerhafte Protokollzustände und unentdeckte Übertragungsfehler zurückgeführt werden. Die Paketzustellrate ist dementsprechend ein geeignetes Kriterium, um die Anpassungsfähigkeiten eines Routingprotokolls zu bewerten.

### Schlussfolgerungen

Ziel dieses Kapitels ist es, die Eignung von positionsbasierten Routingprotokollen zu evaluieren. In Abschnitt 2.1 wurde bereits ausgeführt, dass die meisten dieser Ansätze nur Heuristiken sind und somit keine erfolgreiche Pfadsuche garantieren können. Daraus folgt, dass bei der Simulation eines positionsbasierten Ansatzes die Paketzustellrate Verluste sowohl aufgrund der im vorherigen Abschnitt dargestellten Ursachen als auch aufgrund abgebrochener Pfadsuchen berücksichtigt. Um die anderen Ursachen für einen Paketverlust vollständig zu eliminieren, müssen Übertragungsfehler und fehlerhafte Protokollzustände vermieden beziehungsweise bei der Auswertung berücksichtigt werden. Aus diesem Grund wird im Folgenden auf eine Knotenbewegung während eines Simulationsdurchlaufs verzichtet. Des Weiteren werden Übertragungsfehler ausgeschlossen, indem ein deterministisches Funkwellenausbreitungsmodell verwendet und der Medienzugriff abstrahiert wird.

Der Verzicht auf eine Knotenbewegung während der Simulation bedeutet nicht, dass die Knoten zum Evaluationszeitpunkt nicht realistisch positioniert sein können. Vielmehr geht es bei dieser Entscheidung darum, anhand von Momentaufnahmen zu analysieren, wie zuverlässig die heuristischen Ansätze bei der aktuellen Topologie funktionieren. Da die eigentliche Wegewahl nur einige hundert Millisekunden benötigt, ist es valide, die Bewegung der Knoten während der Dauer einer Suche zu vernachlässigen. Bei der beschriebenen Evaluation von Momentaufnahmen gibt es keinen zeitlichen Ablauf; daher können auch keine Paketkollisionen auftreten. Ein Vorteil dieses Ansatzes ist die hohe Berechnungsgeschwindigkeit, die es ermöglicht, für alle denkbaren Kommunikationspartner eine Pfadsuche durchzuführen. Die Details der verwendeten Simulationsumgebung werden in den folgenden Unterabschnitten erläutert.

#### 3.2.1 Netzwerktopologie und Funkhindernisse

Der entwickelte Simulator evaluiert, wie erwähnt, die Zuverlässigkeit der Wegewahl unterschiedlicher positionsbasierter Verfahren anhand von Momentaufnahmen. Somit bleibt die Frage, welches Bewegungsmodell eingesetzt wird, um die Momentaufnahmen zu erstellen. Da Routingprotokolle für mobile Ad-hoc-Netzwerke möglichst universell einsetzbar sein sollen, werden für Simulationszwecke meist zufallsbasierte Bewegungsmodelle wie MRD herangezogen. Für die Simulation von Ansätzen, die speziell für Fahrzeugnetze entwickelt wurden, werden meist Verkehrssimulatoren bevorzugt.

Die ersten positionsbasierten Verfahren wurden in Netzwerken mit einer statischen Topologie evaluiert. Durch den technischen Fortschritt wurde es immer plausibler, von mobilen Geräten auszugehen. Deshalb wird häufig eine leere Ebene angenommen, auf der die Knoten

verteilt werden und sich nach einem zufälligen Muster bewegen. Da keine weiteren Informationen über den Einsatzzweck vorliegen, ist es plausibel, die Knoten zu Beginn der Simulation gleichmäßig im Simulationsgebiet zu verteilen. Um die Konnektivität zwischen den kommunizierenden Knoten zu gewährleisten, wird die Partitionierung des Netzwerks meist durch die Wahl einer hinreichend hohen Knotendichte vermieden. Bei dieser Herangehensweise gibt es allerdings das Problem, dass in einem dichten und über das gesamte Simulationsgebiet gleichmäßig verteilten Netzwerkgraphen ein starker Zusammenhang zwischen der Strecke vom Start zum Zielknoten und dem kürzesten Pfad zwischen diesen beiden besteht. Daher gelangen die positionsbasierten Heuristiken in derartigen Netzwerken nur selten in ein lokales Optimum und können somit in den meisten Fällen ohne Überbrückungsstrategie einen Pfad zum Ziel finden.

Der Zusammenhang zwischen der Himmelsrichtung, in welcher der Zielknoten liegt, und dem Verlauf eines existierenden Pfads zu diesem hin wird im Wesentlichen durch die folgenden Faktoren beeinflusst:

- **Knotendichte:** In dem betrachteten Szenario steigt mit sinkender Knotendichte das Risiko einer Netzwerkpartitionierung. Dadurch ist es möglich, dass diejenigen Knoten, die in der Richtung des Ziels liegen, zu einer Partitionsgrenze führen, welche nur durch eine Vergrößerung der Entfernung zum Ziel umgangen werden kann (vergleiche Abbildung 2.5).
- **Knotenverteilung:** In vielen Simulationen werden die Knoten in Ermangelung weiterer Informationen über ein reales Einsatzszenario gleichmäßig über das Simulationsgebiet verteilt. Je ungleichmäßiger die Knotenverteilung ist, desto höher ist die Wahrscheinlichkeit, im Vergleich zur Luftlinie Umwege zum Zielknoten in Kauf nehmen zu müssen. Hierdurch steigt die Wahrscheinlichkeit, dass die Heuristiken in ein lokales Optimum geraten und die Pfadsuche abbrechen.
- **Hindernisse:** Auch bei ausreichender Knotendichte und bei einer gleichmäßigen Verteilung kann die Konnektivität durch Hindernisse beeinträchtigt werden, die von der verwendeten drahtlosen Übertragungstechnik nicht überwunden werden können. Befindet sich beispielsweise ein Gebäude zwischen dem Start- und dem Zielknoten, so muss ein Pfad um dieses Hindernis herum gefunden werden. Ein Funkhindernis kann dementsprechend sowohl die Konnektivität zwischen räumlich nahe liegenden Knoten beeinträchtigen, als auch durch seine räumliche Ausdehnung die gleichmäßige Positionierung der Knoten behindern.

Ohne detaillierte Kenntnisse des Einsatzszenarios kann über eine sinnvolle Knotendichte und -verteilung keine Aussage getroffen werden. Deshalb ist die meist herangezogene Annahme einer Gleichverteilung der Knoten noch am plausibelsten. Des Weiteren erscheint es auch sinnvoll, eine Knotendichte zu wählen, die im Zusammenhang mit der verwendeten Funkreichweite zu einer guten Konnektivität innerhalb des Netzwerks führt. Aus diesen Gründen



ist das Hinzuziehen von möglichen Funkhindernissen am besten geeignet, um positionsbasierte Ansätzen auf ihre Eignung hin zu untersuchen. Zusätzlich ist es gerade in dicht besiedelten Ländern wenig plausibel, ein mehrere Quadratkilometer großes Gebiet ohne Bebauung als Simulationsszenario zu verwenden.

Die verwendeten Funkhindernisse können zufällig generiert werden. Allerdings müssen dann bezüglich der genauen Form der Hindernisse und ihrer Häufigkeit genaue Vorgaben gemacht werden. Da die Festlegung dieser Kenndaten aber völlig beliebig wäre, wird im Folgenden dargestellt, wie die Hindernisse anhand der Bebauung einer realen Innenstadt erstellt wurden.

Um zu spezifizieren, an welchen Stellen des Simulationsgebiets sich Hindernisse befinden und wo sich Knoten aufhalten können, wurde die Straßenkarte des *OpenStreetMap*-Projekts [14] herangezogen. Diese Karte wird anhand von freiwilligen Benutzereingaben erstellt und steht unter der *Creative-Commons-Attribution-ShareAlike-2.0-Lizenz* [6]. Im Vergleich zu anderen kostenlosen Kartendiensten bietet OpenStreetMap den großen Vorteil, dass auch die von den Benutzern eingereichten GPS-Daten verfügbar sind und für eigene Zwecke genutzt werden können. Da die Mannheimer Innenstadt zum Zeitpunkt des Datenexports (Ende 2007) noch nicht in OpenStreetMap verfügbar war, wurde das Zentrum von Karlsruhe als Simulationsgebiet ausgewählt.

Der Server von OpenStreetMap liefert die gewünschte Karte in Form einer XML-Datei, in der alle Straßen der angegebenen Region gespeichert sind. Jede Straße wird durch eine Liste von Strecken spezifiziert und diese wiederum als die Kombination von zwei Knoten. Die Knoten werden anhand einer eindeutigen Knotennummer identifiziert und enthalten ihre Position als Breiten- und Längengrad. Zusatzinformationen wie beispielsweise Straßennamen sind ebenfalls in den Straßen-Tags der XML-Datei gespeichert. Die exportierte XML-Datei ist für das lokale Editieren optimiert und enthält deshalb alle zu einer Straße gehörigen Strecken und Knoten. Aus diesem Grund wurden nach dem Export alle vollständig außerhalb des Simulationsgebiets liegenden Strecken und deren zugehörige Knoten entfernt. Bei Strecken, die einen Knoten außerhalb des Simulationsgebiets besitzen, wurde dessen Position durch den Schnittpunkt der zugehörigen Strecke mit der Gebietsbegrenzung ersetzt. Abschließend wurden alle für die Simulation nicht benötigten Daten wie beispielsweise die Beschreibung von Parkanlagen entfernt.

Der Einfluss von Gebäuden auf die Funkwellenausbreitung hängt von zahlreichen Einflussfaktoren ab und ist sehr komplex zu modellieren. So kann es beispielsweise durch Reflexionen an Wänden zu einer Mehrwegausbreitung oder zu einer Wellenüberlagerung kommen. Da derartige Faktoren in einer großflächigen und auf Momentaufnahmen basierenden Simulation nicht akkurat berücksichtigt werden können, werden alle Gebäude als nicht überwindbare Hindernisse modelliert. Eine Strategie, die Hindernisse automatisch zu generieren, besteht darin, diese parallel zum Straßenverlauf auf beiden Seiten im Abstand von wenigen Metern zu erstellen. Dabei müssen die Hindernisse an Straßenkreuzungen mit entsprechenden Aussparungen versehen werden. Diese automatische Vorgehensweise hat allerdings den Nach-



## 3.2 Erstellen einer geeigneten Simulationsumgebung

teil, dass die zahlreichen unbebauten Gebiete wie zum Beispiel die Grünanlagen vor Gebäuden nicht berücksichtigt werden (vergleiche Abbildung 3.4). Ebenfalls sind nicht alle Straßen einspurig und gleich breit. Aus diesen Gründen ist die beschriebene Strategie nur begrenzt geeignet, um die Pfadsuche von positionsbasierten Routingprotokollen anhand einer großflächigen Simulationsstudie zu bewerten.

Mit Hilfe hochauflösender Satelliten- oder Flugzeugfotos können die Umrisse der realen Bebauung einer Stadt mit hoher Genauigkeit erfasst werden. Als Datenquelle wurden die Satellitenfotos von *Google Maps* und *Google Earth* verwendet. Allerdings stimmen die Straßenverläufe von OpenStreetMap und Google Maps aufgrund von Ungenauigkeiten bei der Positionsbestimmung und der Kartenausrichtung nicht exakt überein. Zwar ist es für die Simulation an sich unwesentlich, ob eine Straße zehn Meter versetzt liegt oder nicht, aber die Straßen sollten möglichst genau zu den Hindernissen passen. Deshalb wurde im Rahmen dieser Dissertation eine Webanwendung namens *MapEdit* entwickelt, mit der die Knoten aus OpenStreetMap in Google Maps dargestellt und dort händisch an den Straßenverlauf angepasst werden können. Abbildung 3.3 zeigt einen in MapEdit dargestellten Ausschnitt des Simulationsgebiets. Die Knoten können aus der Liste auf der rechten Seite ausgewählt und direkt in der Karte neu positioniert werden.

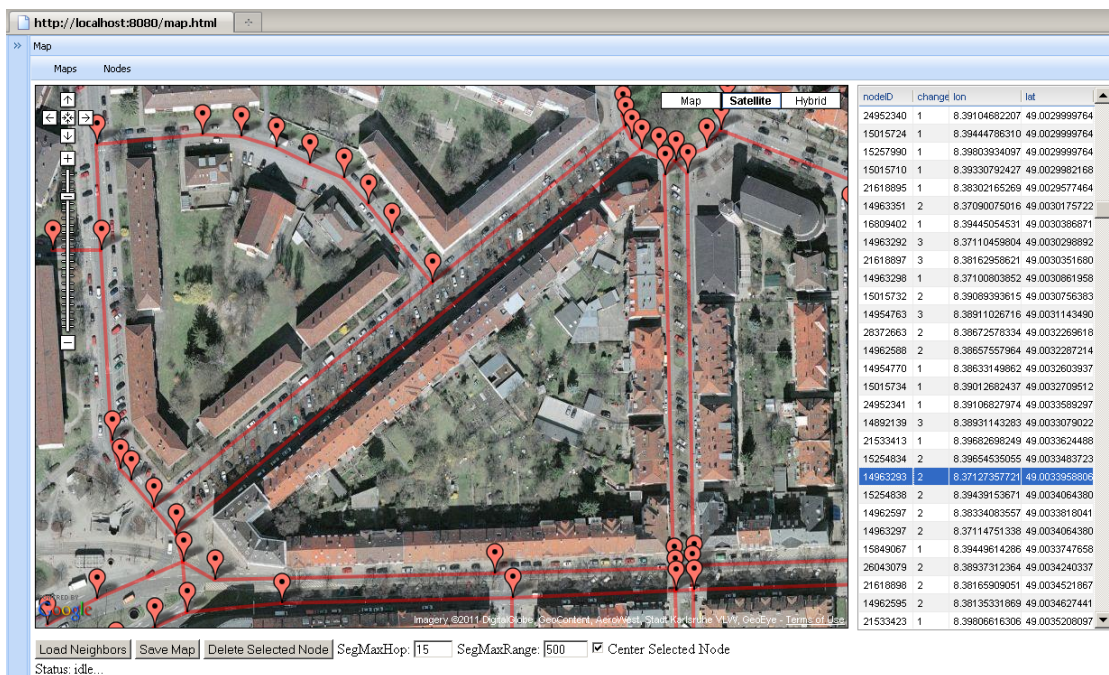


Abbildung 3.3: Screenshot der Webanwendung MapEdit

Nachdem die Knoten an den Straßenverlauf in Google Maps angepasst worden waren, wurde ein Werkzeug zum Nachzeichnen der Gebäudeumrisse benötigt. Allerdings zeigte sich schon bei den Arbeiten an MapEdit, dass die von Google Maps verwendeten Algorithmen zum Zeich-

nen von Linien oder Polygonen für die gleichzeitige Darstellung von sehr vielen geometrischen Objekten ungeeignet sind. Dies liegt weniger an den Algorithmen selbst als an deren Implementierung in JavaScript und der Leistungsfähigkeit der JavaScript-Engines gängiger Webbrowser. Deshalb wurde die digitale Straßenkarte in die *Keyhole Markup Language* [9] überführt, die von Google Earth gelesen und geschrieben werden kann. In Google Earth wurden dann die Umrisse von allen Gebäuden innerhalb des Simulationsgebiets per Hand nachgezeichnet und ebenfalls in der KML-Datei gespeichert. Abbildung 3.4 zeigt das 2,5 km breite und 1,6 km lange Simulationsgebiet in Google Earth. Die Straßen sind hierbei in Weiß und die Hindernisse in Blau eingezeichnet.

Die Straßen der Karte bestehen aus 1046 Straßenabschnitten und 835 Knoten. Da sich aufeinanderfolgende Straßenabschnitte einen Knoten teilen und an Kreuzungen mehrere Straßen an einer Position zusammentreffen, ist die Anzahl der Knoten deutlich kleiner als die der Straßenabschnitte. Im Folgenden werden die 390 Positionen, von denen drei oder mehr Straßen abgehen, als Straßenkreuzungen betrachtet. Das Straßennetz ist circa 65,8 km lang, wobei die einzelnen Spuren von Hauptverkehrsstraßen separat gezählt werden. Des Weiteren wurde die Karte um 2064 Hindernisstrecken ergänzt, die insgesamt eine Länge rund 88 km aufweisen.

### 3.2.2 Funktionsweise und Konfiguration des Simulators

Das Ziel der in dieser Arbeit beschriebenen Simulationsstudie ist es, die Eignung der Pfadsuche von positionsbasierten Routingverfahren in Innenstadtszenarien zu bewerten. Da es sich bei dem hierfür entwickelten Simulator *ConSim* nicht um einen klassischen ereignisbasierten Netzwerksimulator handelt, wird im Folgenden kurz seine Funktionsweise vorgestellt.

Nachdem die Hindernisse aus der KML-Datei in ein von ConSim lesbares Format umgewandelt worden waren, musste ein Mechanismus zur Verteilung der zu simulierenden Netzteilnehmer ausgewählt werden. Da der Straßenverlauf im Simulationsszenario bekannt war, war es naheliegend, einen Bewegungssimulator zu verwenden. Derartige Simulatoren schreiben die generierten Bewegungsdaten typischerweise in Logdateien, die von anderen Programmen weiterverarbeitet werden können. Mit Hilfe dieser Dateien ist es möglich, die Knotenbewegungen nachzuvollziehen und somit Momentaufnahmen für den Netzwerksimulator zu generieren. Diese Strategie wurde im Rahmen dieser Dissertation mit dem freien Simulator SUMO erprobt. Da keine Verkehrsflussinformationen über das Szenario vorlagen, wurde der sogenannte Duarouter verwendet, der ein Fahrzeug auf dem kürzesten Weg von der Start- zur Zielposition bewegt. Zu Beginn wurde jedem Fahrzeug ein zufällig ausgewählter Straßenabschnitt als Zielposition zugewiesen. SUMO simulierte anhand dieser Konfiguration das Verhalten und die Bewegung der einzelnen Verkehrsteilnehmer. In allen durchgeführten Simulationen traten allerdings schon nach kurzer Zeit starke Verkehrsstaus auf. Die Ursache hierfür liegt in dem Umstand, dass die Rechts-vor-links-Regel alleine nicht ausreicht, um den gesamten Straßenverkehr einer Stadt effizient zu regulieren. Zwar bietet SUMO die Möglichkeit, Ampelanlagen zu simulieren, allerdings müssen dafür die Ampelphasen manuell angegeben werden. Aus diesen Gründen ist eine realistische Verkehrssimulation ohne detailliertes Wissen über den tatsäch-





Abbildung 3.4: Das ca. 4 km<sup>2</sup> große Simulationsgebiet in Google Earth.

lichen Verkehrsfluss und die eingesetzten Verkehrssteuerungsmaßnahmen nicht möglich. Da diese Daten weder vorlagen noch mit adäquatem Aufwand beschafft werden konnten, wurden die Knoten gleichmäßig über die Straßen verteilt.

Da alle Verfahren anhand der gleichen Knotenverteilung evaluiert werden sollten, wird die Verteilung von einem separaten Programm berechnet. Dieses liest die Hindernisse und Straßendaten ein, wandelt deren Positionen mit einer auf dem WGS-84-Ellipsoid [114] basierenden Mercator-Projektion in kartesische Koordinaten um und führt anschließend eine geeignete Translation des Koordinatensystems durch. Im nächsten Schritt werden so lange Knoten auf den Straßen verteilt, bis die angegebene Dichte erreicht ist. Da die OpenStreetMap-Daten den Straßentyp enthalten, kann bei dieser Operation zwischen Neben- und Hauptstraßen unterschieden werden. Im Folgenden wird auf den Hauptstraßen immer eine doppelt so hohe Knotendichte wie auf den Nebenstraßen verwendet. Nachdem die jeweilige Verteilung berechnet worden ist, bestimmt das Programm anhand der angegebenen Funkreichweite und der Funkhindernisse den Netzwerkgraphen  $G = (V, E)$ . Dabei wird für jeden Knoten  $v_1 \in V$  eine Kante zu allen Knoten  $v_2 \in V \setminus \{v_1\}$  erzeugt, wenn der euklidische Abstand  $d(v_1, v_2)$  kleiner oder gleich der Funkreichweite ist und der Vektor  $\overrightarrow{v_1 v_2}$  kein Hindernis schneidet. Da die Pfadsuche nur zwischen miteinander verbundenen Knoten erfolgreich sein kann, werden – noch während das Szenario erstellt wird – die Zusammenhangskomponenten des Graphen bestimmt. Abschließend speichert das Programm die gefundenen zusammenhängenden Teilgraphen und alle anderen relevanten Daten in einer XML-Datei.

Wie in Abschnitt 2.1 erläutert, wählen die gängigen positionsbasierten Verfahren den nächsten Weiterleiter anhand von lokalem Wissen aus. Neben den Positionen des Ziels und des aktuellen Knotens werden direkt (Paketaustausch) oder indirekt (verteilter Wettbewerb) auch die Positionen der benachbarten Knoten innerhalb der Funkreichweite berücksichtigt. Deshalb generiert ConSim für jeden Knoten des Szenarios eine Nachbarschaftstabelle, die von den implementierten Verfahren für die Pfadsuche verwendet werden kann. Anhand dieser vorab berechneten Datenstrukturen kann ein Pfad im Vergleich zu einer klassischen ereignisbasierten Simulation sehr schnell berechnet werden. Dieser Umstand ermöglicht es, für jeden zusammenhängenden Teilgraphen  $G_t = (V_t, E_t)$  des aktuellen Netzwerks alle geordneten Knotenpaare  $\{(v_1, v_2) \mid v_1 \in V_t, v_2 \in V_t, v_1 \neq v_2\}$  zu berücksichtigen. Der Simulator berechnet dementsprechend für jedes implementierte Routingverfahren und jedes Knotenpaar  $(v_1, v_2)$  eines Teilgraphen einen Pfad von  $v_1$  nach  $v_2$ . Ein verwendetes Szenario besteht beispielsweise aus 1670 Knoten in 8 Zusammenhangskomponenten. In diesem Beispiel sucht ConSim pro Routingverfahren nach 2,6 Millionen Pfaden. Nach der Auswertung aller Teilgraphen werden die Simulationsergebnisse der einzelnen Protokolle jeweils in einer Datei hinterlegt. Soweit nicht anders angegeben, basieren alle im Folgenden präsentierten Ergebnisse auf den Resultaten von jeweils zehn Szenarien.



### 3.2.3 Definition der Bewertungskriterien

Die berechneten Pfade zwischen allen möglichen Kommunikationspartnern werden pro Szenario und Routingprotokoll in einer komprimierten Datei gespeichert und können sowohl zur Laufzeit als auch zu einem späteren Zeitpunkt ausgewertet werden. Dabei werden auch die unvollständigen Pfade von abgebrochenen Suchvorgängen erfasst.

Die Menge  $P_{A,B}$  enthält die durch das Routingverfahren  $A$  bestimmten Pfade aller Szenarien mit der Konfiguration  $B$ . Die Schwierigkeit, einen Pfad zu finden, hängt unter anderem vom räumlichen Abstand der beiden Kommunikationspartner zueinander ab. Aus diesem Grund liegt es nahe, die Simulationsergebnisse nach dem räumlichen Abstand der jeweiligen Knoten zueinander zu gruppieren. In einem Innenstadtszenario ist es allerdings möglich, dass Funkhindernisse auch bei vergleichsweise nah beieinander liegenden Knoten zu größeren Umwegen führen und somit den Zusammenhang zwischen Komplexität der Pfadsuche und räumlichem Abstand abschwächen. Daher werden im Folgenden die gefundenen Pfade  $P_{A,B}$  meist nach der Länge des kürzesten Pfades zwischen den jeweiligen Kommunikationspartnern gruppiert. Je nach Bedarf wird eine Gruppierung nach den kürzesten Pfaden bezogen auf die Anzahl der Hops (Kantengewicht von eins) oder bezogen auf die Pfadlänge (Kantengewicht abhängig vom räumlichen Abstand der Knoten) herangezogen.

Die Pfade aller erfolgreichen Suchvorgänge sind in  $PE_{A,B} \subset P_{A,B}$  enthalten. Das wichtigste Kriterium zur Bewertung eines Verfahrens ist der Prozentsatz der erfolgreich gefundenen Pfade  $GP = |PE_{A,B}|/|P_{A,B}|$ . Dieser gibt also im Detail an, für welchen Prozentsatz der möglichen Kommunikationspartner aller Simulationsdurchläufe das betrachtete Protokoll auch tatsächlich einen Pfad gefunden hat. Da die Menge der möglichen Kommunikationspartner stark von der Knotendichte und der Funkreichweite beeinflusst wird, beziehen sich die Prozentsätze verschiedener Konfigurationen auf unterschiedliche Datensätze und können somit nur bedingt miteinander verglichen werden. Bei einer kleineren Funkreichweite ist beispielsweise die Anzahl der Hops der kürzesten Pfade durchgehend größer als bei einer größeren Funkreichweite bei gleicher Knotendichte. Zum Vergleich unterschiedlicher Konfigurationen werden daher die Ergebnisse bevorzugt nach der räumlichen Pfadlänge gruppiert.

Ein weiteres Bewertungskriterium ist die Länge der gefundenen Pfade. Neben der durchschnittlichen absoluten Pfadlänge pro Gruppierungsmerkmal kann die Länge auch im Verhältnis zum jeweils kürzesten Pfad ausgedrückt werden. Des Weiteren sind für die Analyse der Überbrückungsstrategien zusätzliche Bewertungs- und Filterkriterien von Bedeutung. Hierzu zählt beispielsweise die Angabe, wie oft eine Überbrückung durchschnittlich notwendig war und wie viele Hops pro Überbrückung benötigt wurden. Zudem kann bei der Längenberechnung zwischen den Pfadabschnitten im Haupt- und im Überbrückungsverfahren unterschieden werden. Außerdem werden Filter eingesetzt, um zu analysieren, wie viel Prozent der Pfade gefunden werden, wenn im Überbrückungsverfahren maximal  $x$  Hops pro Überbrückung erlaubt sind. Dieser Wert ist beispielsweise von Bedeutung, um den Suchradius der distanzvektorbasierten Überbrückung zu konfigurieren.

### 3.3 Simulationsergebnisse bestehender Verfahren

Wie in Abschnitt 3.2.1 beschrieben, hängt der Erfolg der Pfadsuche bei Greedy-Algorithmen wie GRS in der häufig verwendeten freien Ebene ohne Funkhindernisse primär von der Knotendichte und -verteilung ab. Es ist plausibel, dass die Routingheuristiken für nahezu alle Kommunikationspartner einen Pfad finden, sobald die Knotendichte ein bestimmtes Maß übersteigt. Somit sind sie nur im Ausnahmefall auf eine Überbrückungsstrategie angewiesen. Allerdings ist ein mehrere Quadratkilometer großes Einsatzgebiet ohne Funkhindernisse in den dicht besiedelten Industriestaaten eher unwahrscheinlich, sodass für die Simulationsstudie ein Innenstadtszenario mit Funkhindernissen ausgewählt wurde. Bevor allerdings die eigentlichen Ergebnisse für das Innenstadtszenario beschrieben werden, wird der neu entwickelte Simulator dazu verwendet, den vorab beschriebenen Zusammenhang zwischen Knotendichte und Pfadsuche in der freien Ebene zu untersuchen.

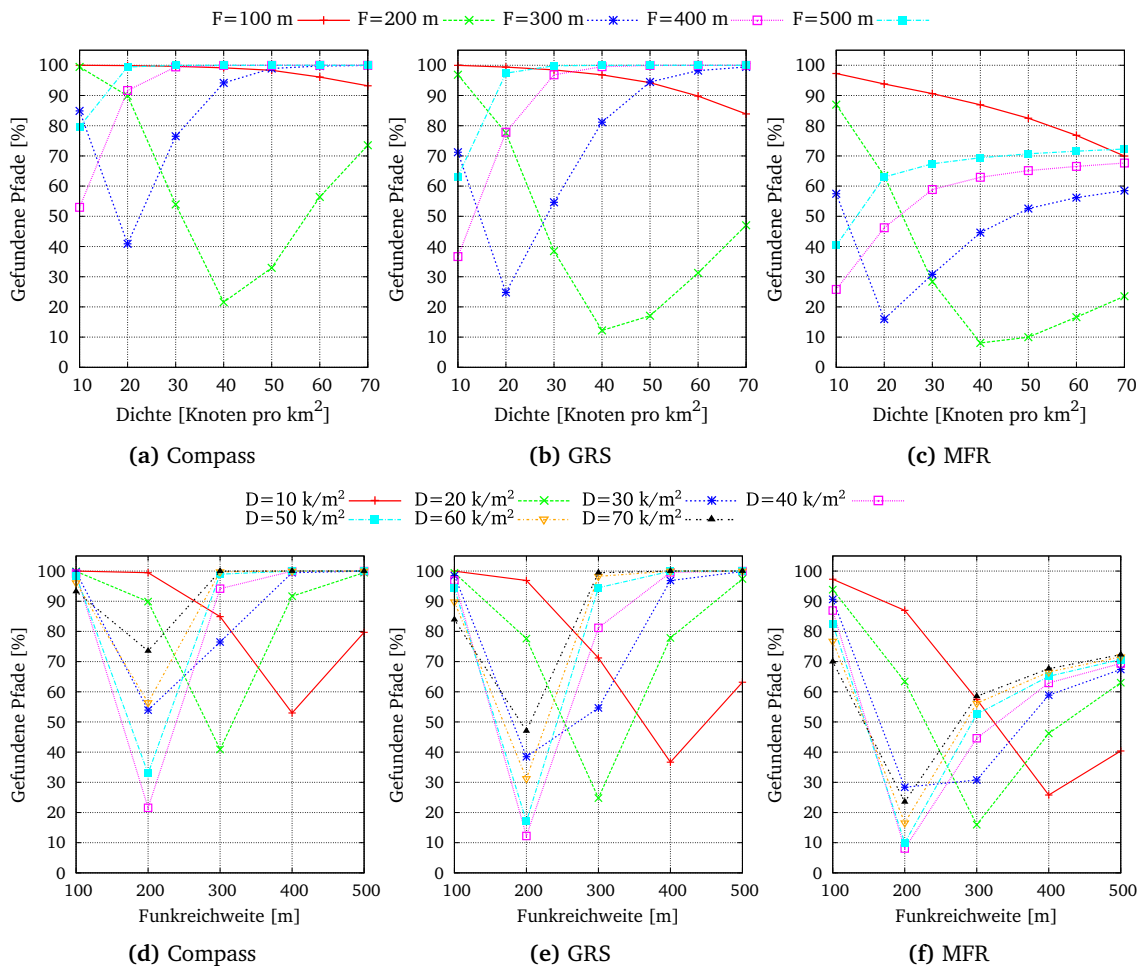
#### 3.3.1 Knotendichte und Knotenverteilung

In diesem Abschnitt werden die Ergebnisse einer Simulationsstudie dafür verwendet, die Hypothesen, die zu der Gestaltung des Simulationsszenarios geführt haben, empirisch zu untermauern. Hierfür wurden in einem 5 km x 5 km großen Gebiet die Knoten gleichmäßig in der Ebene verteilt. Als Knotendichte wurde 10-70 Knoten pro Quadratkilometer und als Funkreichweite 100-500 m gewählt. Da in dieser Simulation die konkrete Knotenverteilung einen großen Einfluss hat, wurde jede der 35 Kombinationen mit 50 unterschiedlichen Knotenverteilungen simuliert. Die Ergebnisse für Compass, GRS und MFR werden in Abbildung 3.5 dargestellt. Es wurde auch das Compass-Verfahren auf dem *Gabriel Graph* (GG) berechnet. Da sich die Ergebnisse allerdings von denen der Variante ohne Planarisierung nur unwesentlich unterscheiden, wird hier aus Platzgründen auf eine Präsentation verzichtet. Auf der x-Achse sind die Knotendichte beziehungsweise die Funkreichweite und auf der y-Achse der Prozentsatz der Kommunikationspartner, für die ein Pfad gefunden wurde, dargestellt. Jeder Punkt im Diagramm entspricht den zusammengefassten Ergebnissen der jeweils 50 Simulationsdurchläufe. Dabei ist zu beachten, dass sowohl die Anzahl der möglichen Kommunikationspartner als auch deren räumlicher Abstand zueinander stark von der Knotendichte und der Funkreichweite abhängen und sich somit die Prozentsätze der gefunden Pfade auf unterschiedliche Topologien beziehen.

Auch wenn sich der konkrete Verlauf der drei Verfahren in Abbildung 3.5 unterscheidet, ist dennoch ein gemeinsames Muster zu erkennen. Ausgehend von einer konstanten Funkreichweite von 400 m oder mehr, führt eine Steigerung der Knotendichte zu einer Erhöhung des Prozentsatzes an gefunden Pfaden. Bei den Funkreichweiten von 200 m und 300 m kommt es vor einer Steigerung des Prozentsatzes zuerst zu einem deutlichen Abfall. Dieser begründet sich durch den Umstand, dass die betreffenden Funkreichweiten bei einer sehr niedrigen Knotendichte nicht ausreichen, um alle Knoten im Simulationsgebiet miteinander zu verbinden. Dementsprechend sind nahezu alle möglichen Kommunikationspartner räumlich benachbart und mit wenigen Hops zu erreichen. Aus diesem Grund finden die Routingheuristiken in die-

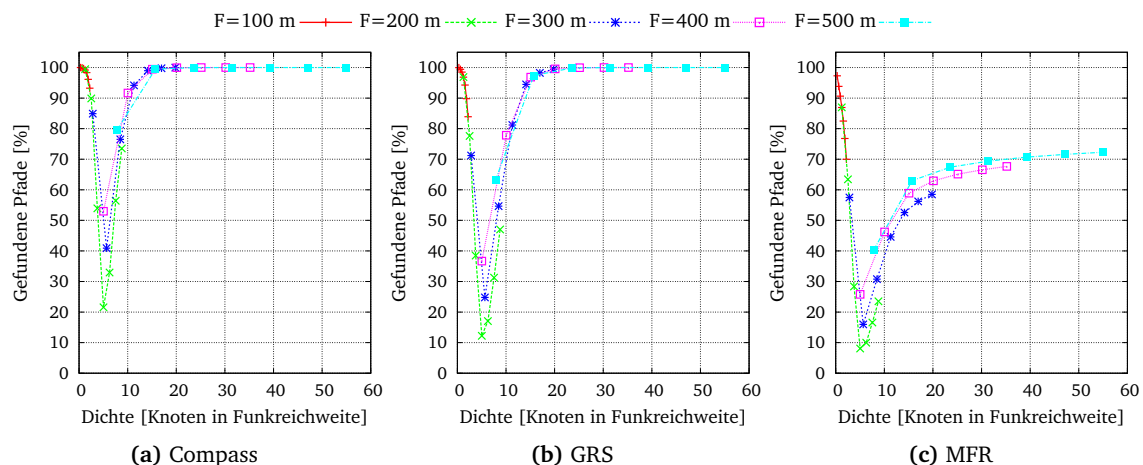
### 3.3 Simulationsergebnisse bestehender Verfahren

sen Szenarien nahezu alle möglichen Pfade. Mit steigender Knotendichte können immer mehr Knoten miteinander kommunizieren, allerdings nur über einen vergleichsweise dünnen Netzwerkgraphen. Dieser Umstand führt zu einer deutlichen Reduktion der gefundenen Pfade. Alle drei Verfahren erreichen ihren Tiefpunkt bei 20 bzw. 40 Knoten pro  $\text{km}^2$ . GRS kann beispielsweise bei einer Funkreichweite von 200 m und einer Knotendichte von 40 Knoten pro  $\text{km}^2$  nur noch circa 11 % aller existierenden Pfade finden. Im direkten Vergleich dazu kann das Compass-Verfahren noch fast doppelt so viele Pfade finden, wohingegen MFR mit rund 8 % schlechter als GRS abschneidet. Bei einer Funkreichweite von 100 m ist der Bereich der untersuchten Knotendichten nicht ausreichend, um den Tiefpunkt zu erkennen. Analog zu dieser Situation ist bei einer Funkreichweite von 400 m oder höher nur noch der wieder ansteigende Kurvenverlauf zu sehen.



**Abbildung 3.5:** Simulationsergebnisse der Routingheuristiken in der freien Ebene ohne Funkhindernisse sowie mit unterschiedlichen Knotendichten und Funkreichweiten.

Die zweite Zeile von Abbildung 3.5 zeigt die Ergebnisse der Simulationen in einer alternativen Darstellung. Während die Funkreichweite auf der x-Achse dargestellt ist, sind die Knotendichten pro Kurvenverlauf konstant. Alle Knotendichten von mehr als 20 Knoten pro  $\text{km}^2$  haben ihren Tiefpunkt bei einer Funkreichweite von 200 m und weisen einen ähnlichen Kurvenverlauf auf. Wird die Funkreichweite verdoppelt, so vervierfacht sich das Kommunikationsgebiet eines Knotens. Wird gleichzeitig das Simulationsgebiet um den Faktor vier vergrößert und die Knotendichte auf ein Viertel gesenkt, so ergibt sich wieder das ursprüngliche Szenario in einem anderen Maßstab. Auch wenn in der durchgeführten Studie die Abmessungen des Simulationsgebietes konstant waren, lässt sich dieser Zusammenhang dennoch an den Ergebnissen ablesen. So ist beispielsweise der Prozentsatz der gefundenen Pfade bei einer Knotendichte von 40 Knoten pro  $\text{km}^2$  und einer Funkreichweite von 200 m nahezu identisch zu dem Datensatz mit einem Viertel der Knotendichte und doppelter Funkreichweite. Die eigentlich relevante Einflussgröße ist somit die durchschnittliche Anzahl gleich verteilter Knoten innerhalb der jeweiligen Funkreichweite.



**Abbildung 3.6:** Simulationsergebnisse der Routingheuristiken in der freien Ebene ohne Funkhindernisse. Die Knotendichten und die Funkreichweiten der einzelnen Simulationsläufe wurden zu „Knoten in Funkreichweite“ kombiniert.

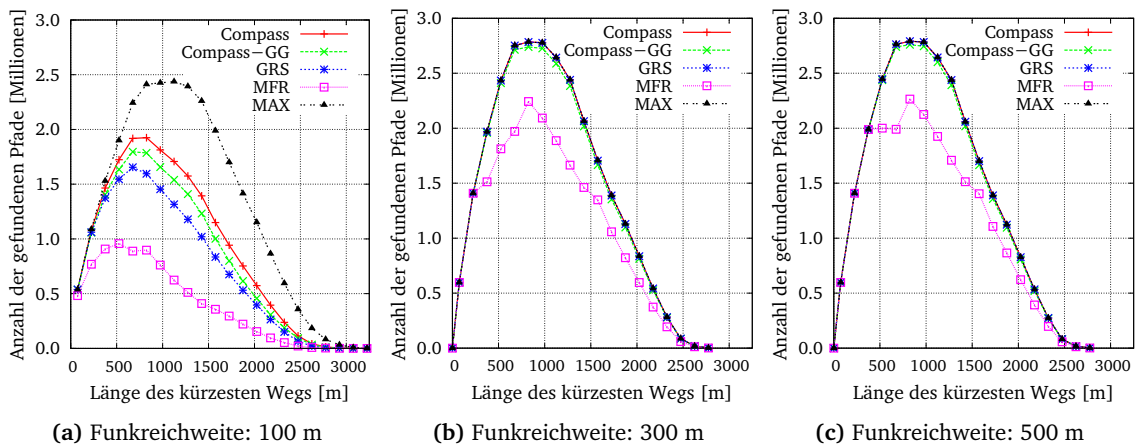
Abbildung 3.6 zeigt die Simulationsergebnisse mit umgerechneter Knotendichte. Ausgehend von einer konstanten Funkreichweite, ist auf der x-Achse die durchschnittliche Anzahl der Knoten innerhalb der jeweiligen Funkreichweite angegeben. Auch wenn nicht alle Kurven den gesamten Wertebereich abdecken, ist durch die Überlappungen der einzelnen Verläufe der Einfluss der Knotendichte innerhalb der Funkreichweite doch gut zu erkennen. Bei bis zu zwei direkten Nachbarn im Durchschnitt können sowohl das Compass-Verfahren als auch GRS nahezu für alle Kommunikationspartner einen Pfad finden. Im weiteren Verlauf sinkt allerdings der Prozentsatz der gefundenen Pfade bei beiden Ansätzen deutlich und erreicht den jeweiligen Tiefpunkt bei etwa fünf direkten Nachbarn. Ab circa 16 beziehungsweise 20 Knoten innerhalb



### 3.3 Simulationsergebnisse bestehender Verfahren

der Funkreichweite können beide Verfahren wieder nahezu alle Pfade finden. Durch die Fokussierung auf den räumlichen Fortschritt in Bezug auf die direkte Strecke zwischen dem Start- und dem Zielknoten erreicht das MFR-Verfahren deutlich schlechtere Ergebnisse und setzt zugleich eine wesentlich höhere Knotendichte voraus. Durch die hier präsentierten Ergebnisse konnte die Hypothese bestätigt werden, dass ab einer hinreichend hohen Knotendichte die positionsbasierten Routingheuristiken ohne Funkhindernisse nahezu immer einen Pfad finden.

Wie bereits in Abschnitt 3.2.1 beschrieben, ist neben der durchschnittlichen Dichte der Knoten auch deren konkrete Verteilung innerhalb des Simulationsgebiets von Bedeutung. Um diesen Sachverhalt anhand empirischer Daten zu belegen, wurde eine weitere Simulationsreihe ohne Funkhindernisse durchgeführt. In Ermangelung einer plausiblen Verteilungsfunktion wurden etwa 25 Knoten pro km gleich verteilt auf den Straßen platziert, wodurch sie automatisch in Bezug auf die gesamte freie Fläche ungleichmäßig verteilt wurden. Im Unterschied zu den Simulationen in den folgenden Abschnitten wurden hier, wie erwähnt, die Umrisse der Gebäude nicht berücksichtigt. Die Ergebnisse für Funkreichweiten von 100 m, 300 m und 500 m werden in Abbildung 3.7 dargestellt. Die Schwierigkeit, einen Pfad zu finden, hängt von dem räumlichen Abstand der beiden beteiligten Knoten ab. Des Weiteren sind sowohl die Anzahl als auch die Länge der Pfade zwischen allen möglichen Kommunikationspartnern nicht für alle Konfigurationen identisch. Deshalb wurden die Ergebnisse entlang der x-Achse nach der Länge des kürzesten Pfades zwischen den jeweiligen Kommunikationspartnern gruppiert. Im Unterschied zu den vorherigen Abbildungen wird hier auf der y-Achse nicht der Prozentsatz der gefundenen Pfade angegeben, sondern deren absolute Anzahl. Diese Darstellungsform ist für diese Simulation sinnvoll, da in ihr die Anzahl der Kommunikationspartner, für die maximal ein Pfad gefunden werden kann, stark von der Funkreichweite abhängt. Aus diesem Grund wurde in den folgenden Abbildungen für die Bewertung der einzelnen Ergebnisse die jeweilige Obergrenze (MAX) ebenfalls eingezeichnet.



**Abbildung 3.7:** Simulationsergebnisse der Routingheuristiken in der freien Ebene bei ungleichmäßiger Knotenverteilung (Knotendichte: 25,37 k/km, keine Funkhindernisse).

Aufgrund der ungleichmäßigen Knotenverteilung und der daraus resultierenden Netzpartitionierung können die Routingverfahren bei einer Funkreichweite von 100 m nur einen Teil aller Pfade finden. Der direkte Vergleich der maximalen Pfadanzahl in den Abbildungen 3.7a und 3.7b zeigt, dass die Funkreichweite von 100 m noch nicht alle Knoten miteinander verbindet und sich somit die Ergebnisse auf mehrere Teilgraphen verteilen. Ab einer Funkreichweite vom 300 m werden viele der knotenfreien Gebiete überbrückt, sodass in dem resultierenden Netzwerkgraphen die meisten Routingheuristiken wieder nahezu alle Pfade finden. In die Abbildungen wurde ebenfalls eine Variante des Compass-Verfahrens (Compass-GG) eingezeichnet, die den Gabriel Graph zur Planarisierung verwendet. Allerdings erzielt Compass-GG aufgrund des weniger dichten planaren Graphen geringfügig schlechtere Resultate und findet im Durchschnitt deutlich längere Pfade als die Variante ohne Planarisierung. Wie in den vorherigen Simulationen schneidet auch hier MFR am schlechtesten ab und kann selbst bei einer Funkreichweite von 500 m für einen großen Anteil von Kommunikationspartnern keinen Pfad finden.

Die hier erläuterten Ergebnisse bestätigen die in Abschnitt 3.2.1 beschriebenen Überlegungen bezüglich der Einflussfaktoren Knotendichte und -verteilung. Da allerdings Compass und GRS schon ab 16 beziehungsweise 20 Knoten innerhalb der Funkreichweite nahezu alle Pfade finden, tritt diese Problematik nicht in allen Simulationsstudien auf. Eine niedrigere Knotendichte ist aber meist auch nicht sinnvoll, da bei ihr das Risiko einer Partitionierung des Netzwerks besteht. Folglich ist in Ermangelung einer plausiblen Knotenverteilung die Berücksichtigung von Funkhindernissen der brauchbarste Ansatz, um die reale Eignung von positionsbasierten Routingverfahren zu evaluieren.

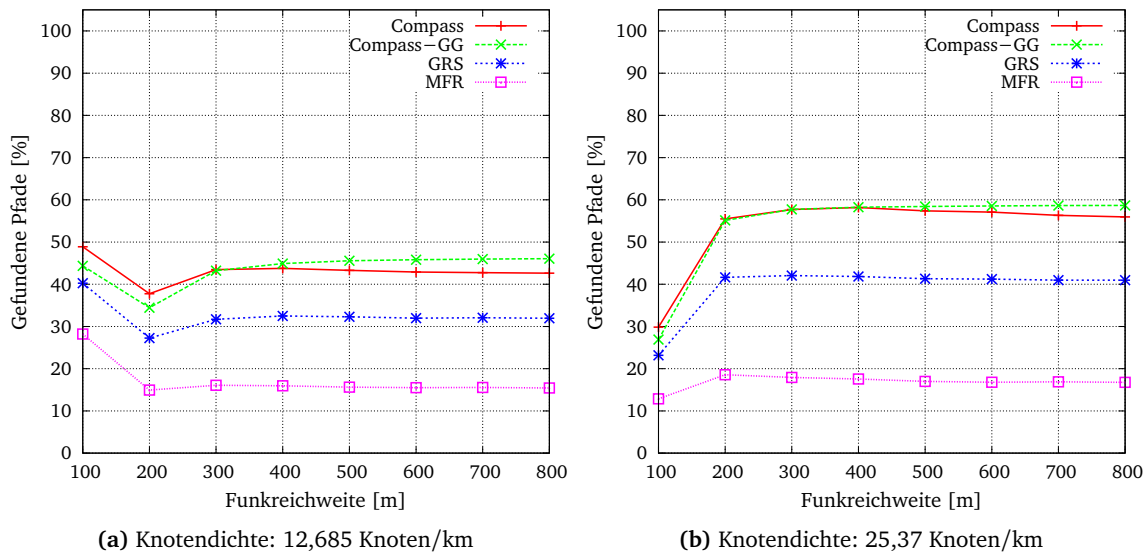
### 3.3.2 Greedy-Algorithmen

Um die Leistungsfähigkeit von positionsbasierten Routingheuristiken zu bewerten, wurde das Innenstadtszenario mit einer Funkreichweite von 100 m bis 800 m und mit Funkhindernissen simuliert. Um auch den Einfluss der Knotendichte auf die Ergebnisse zu untersuchen, wurden Szenarien mit 10 und 20 Knoten/km evaluiert. Der Straßenverlauf des Simulationsgebiets setzt sich aus 48,1 km Nebenstraßen und 17,7 km Hauptstraßen zusammen. Wie bereits in Abschnitt 3.2.2 beschrieben, wird auf den Hauptstraßen immer eine doppelt so hohe Knotendichte wie auf den Nebenstraßen angenommen. Ausgehend von 10 beziehungsweise 20 Knoten pro Kilometer auf den Nebenstraßen ergibt sich eine durchschnittliche Knotendichte von 12,685 beziehungsweise 25,38 Knoten pro Kilometer. Dementsprechend umfassen die Szenarien mit der niedrigeren Knotendichte 835 Knoten und die mit der höheren Dichte 1670 Knoten. Für jedes Routingverfahren wurden die Ergebnisse von vierzig Szenarien mit niedriger Knotendichte gemittelt. Da es bei 1670 Knoten schon circa 2,6 Millionen mögliche Paare von Kommunikationspartnern gibt, war es für diese Konfiguration ausreichend, zehn Szenarien pro Routingverfahren auszuwerten.

Abbildung 3.8 zeigt die Ergebnisse der Routingheuristiken in Abhängigkeit von der Funkreichweite. Bei Reichweiten unter 300 m ist die Konnektivität des Netzwerks noch einge-

### 3.3 Simulationsergebnisse bestehender Verfahren

schränkt. Deshalb kommt es zu Beginn zu einem ähnlichen Kurvenverlauf wie bei den Ergebnissen ohne Funkhindernisse (siehe Abbildung 3.5). Ab einer Funkreichweite von 300 m sind in beiden Konfigurationen nahezu alle Knoten eines Szenarios miteinander verbunden und können als Kommunikationspartner fungieren. Während das Compass-Verfahren bei niedrigerer Knotendichte ab dieser Funkreichweite um die 45 % aller Pfade findet, liegt der Prozentsatz bei GRS und MFR mit 32 % beziehungsweise 15 % deutlich niedriger. Sowohl Compass als auch GRS können in den Szenarien mit höherer Knotendichte das Ergebnis um etwa zehn Prozentpunkte steigern.



**Abbildung 3.8:** Prozentsatz der gefundenen Pfade bei unterschiedlichen Knotendichten und Funkreichweiten.

In dem Szenario mit niedrigerer Knotendichte hat ein Knoten bei einer Funkreichweite von 300 m im Durchschnitt rund 18 direkte Nachbarn. Obwohl diese Kennzahl bei einer Funkreichweite von 600 m auf 28 Knoten ansteigt, erhöht sich der Prozentsatz der gefundenen Pfade bei allen Verfahren nur noch unwesentlich. Bei einer Funkreichweite von 200 m hat ein Knoten in dem Szenario mit höherer Dichte im Durchschnitt auch circa 28 Nachbarn. Dennoch unterscheiden sich die jeweiligen Ergebnisse bei Compass und GRS deutlich. Demnach beeinflussen weder eine höhere Funkreichweite noch eine steigende Anzahl von Knoten innerhalb der Funkreichweite das Ergebnis signifikant. Die Ursache hierfür liegt in der planaren Topologie des Straßennetzes. Nur wenn sowohl auf der jeweiligen Kreuzung als auch auf den angrenzenden Straßen ausreichend viele Knoten vorhanden sind, kann die grundlegende Richtung während der Pfadsuche geändert werden. Im Unterschied zu vorherigen Szenarien kann daher eine niedrigere Knotendichte nicht durch eine erhöhte Funkreichweite kompensiert werden. Dementsprechend hat die Knotendichte den wesentlichsten Einfluss auf die Konnektivität der Knoten und somit auch auf die Anzahl der gefundenen Pfade.

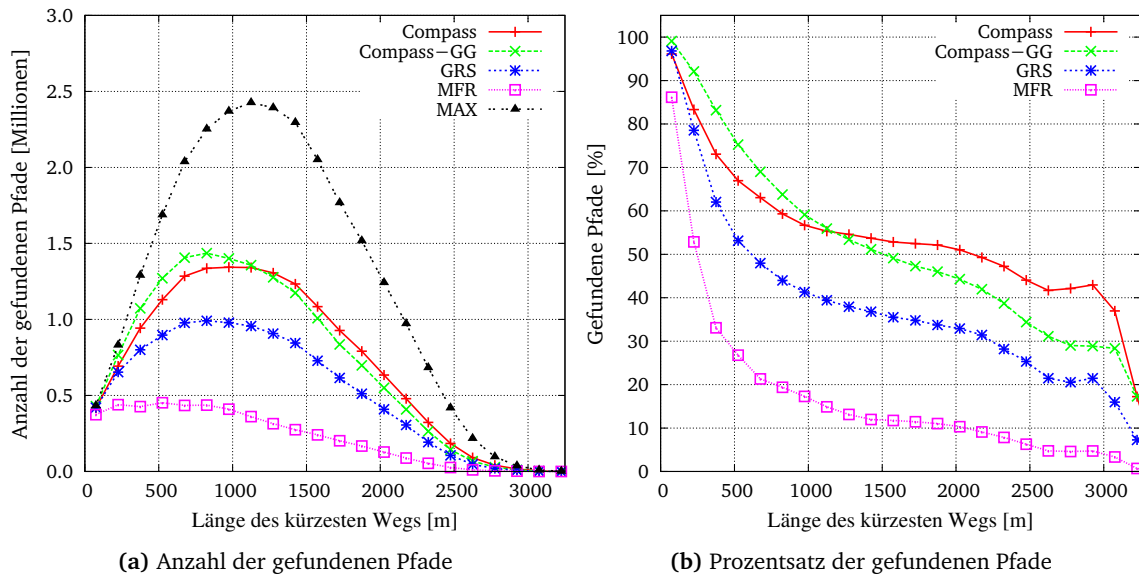
In den dichteren Szenarien finden Compass und GRS für 60 % beziehungsweise 40 % aller Kommunikationspartner einen Pfad. Diese Prozentsätze sind allerdings nicht ausschließlich von der Topologie der Straßen, den Funkhindernissen und der Knotendichte abhängig, sondern auch von der Größe des Simulationsgebiets. Zum einen würde einer Vergrößerung – bei gleicher Knotendichte – zu insgesamt mehr Knoten führen, zum anderen würde sich aber auch die durchschnittliche Länge der kürzesten Pfade zwischen den Kommunikationspartnern vergrößern. Da die Komplexität der Pfadsuche maßgeblich von der Länge des Pfads abhängt, würde eine Vergrößerung des Simulationsgebiets somit zu einer Verschlechterung der Ergebnisse führen.

Aufgrund der gemeinsamen Nutzung des Mediums und der grundsätzlichen Funktionsweise der Datenweiterleitung in einem drahtlosen Ad-hoc-Netzwerk ist der räumliche Abstand von Kommunikationspartnern allerdings ohnehin limitiert. Wenn  $n$  die Anzahl der Knoten des Ad-hoc-Netzwerks und  $W$  der maximal nutzbare Datendurchsatz bei einer konstanten Funkreichweite sind, so beträgt die zu erwartende Bandbreite für jeden Knoten bei jeweils einem zufällig ausgewählten Zielknoten  $\Theta(W/\sqrt{n \log n})$  [67]. Von Sonderfällen abgesehen ist es auch ohne diese formale Beschreibung offensichtlich, dass es für den Datenaustausch zwischen sehr weit voneinander entfernten Knoten besser geeignete Kommunikationsstrukturen als ein drahtloses Ad-hoc-Netzwerk gibt. Wird bei der Pfadsuche allerdings von vorneherein die Anzahl der Hops limitiert, so reduziert sich in der angegebenen Formel die Anzahl der Netzteilnehmer  $n$  auf die jeweilige Anzahl der Knoten innerhalb des Suchradius. Welcher Such- und Kommunikationsradius für den Erfolg der Pfadsuche des jeweiligen Routingverfahrens geeignet ist, wird – soweit es sinnvoll ist – in den jeweiligen Abschnitten untersucht. Die in diesem Abschnitt besprochenen Routingheuristiken ohne Überbrückungsstrategien finden allerdings so wenige Pfade, dass auch mit einer derartigen Limitierung kein sinnvoller Einsatz möglich ist.

Abbildung 3.9 zeigt den Erfolg der Pfadsuche in Abhängigkeit von der Länge des kürzesten Pfads zwischen den jeweiligen Kommunikationspartnern. Dabei wurde links die absolute Anzahl der gefunden Pfade eingezeichnet und rechts der resultierende Prozentsatz. Die Ergebnisse beziehen sich auf eine Funkreichweite von 300 m und eine Knotendichte von 25,37 Knoten pro Kilometer. Von den insgesamt circa 26 Millionen ausgewerteten Paaren von Kommunikationspartnern besitzen etwa 2,5 Millionen einen kürzesten Pfad mit einer Länge zwischen 1050 m und 1200 m. Da die Anzahl der Pfade pro Gruppierungsintervall nicht konstant ist, beziehen sich die angegebenen Prozentsätze auf jeweils unterschiedlich viele Datensätze. Ab einer kürzesten Pfadlänge von etwa 2500 m sinkt die Anzahl der zugrunde liegenden Datensätze sehr deutlich, sodass diese Ergebnisse nur noch bedingt aussagekräftig sind.

Vergleichsweise kurze Pfade werden von allen Verfahren noch zu einem hohen Prozentsatz gefunden. Mit steigendem Abstand sinkt allerdings die Quote kontinuierlich ab. So findet beispielsweise das Compass-Verfahren auf dem Gabriel Graph bei einer Pfadlänge um die 750 m noch circa 60 % aller Pfade, bei einer Pfadlänge um die 2500 m aber nur noch etwa 35 %. Da GRS und MFR sogar deutlich schlechtere Ergebnisse erzielen, sind somit alle hier unter-

### 3.3 Simulationsergebnisse bestehender Verfahren

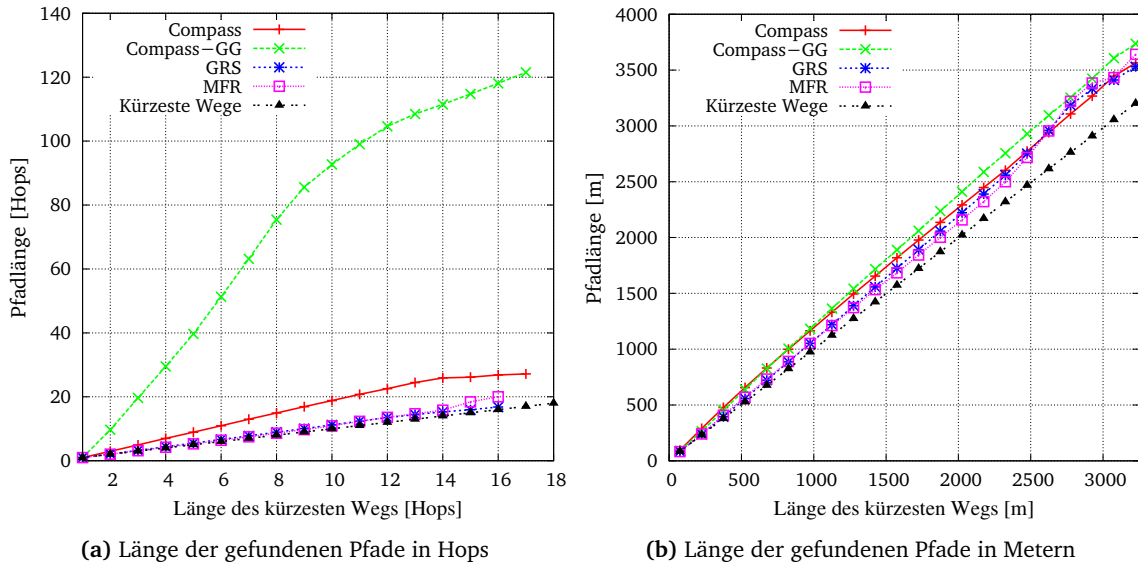


**Abbildung 3.9:** Ergebnisse der Pfadsuche bei einer Funkreichweite von 300 m und einer Knotendichte von 25,37 Knoten/km.

suchten Routingheuristiken für den Einsatz in einem Innenstadtbereich mit Funkhindernissen ungeeignet.

Um die Resultate zu verbessern, können die Basisverfahren durch Überbrückungsstrategien ergänzt werden. Diese werden immer nur dann eingesetzt, wenn das Basisverfahren keinen Pfad finden kann. Aus diesem Grund ist die Güte der in diesem Abschnitt gefundenen Pfade auch für den Einsatz der Überbrückungsstrategien von Bedeutung. Zur Bewertung ihrer Qualität wird in Abbildung 3.10 die Länge der gefundenen Pfade mit den jeweils kürzesten Pfaden verglichen. Dabei sind auf der linken Seite die Ergebnisse in Hops und auf der rechten Seite in Metern dargestellt. Die kürzesten Pfade wurden nach dem jeweiligen Optimierungskriterium der Abbildung (Hops beziehungsweise Meter) berechnet und gruppiert.

In diesem Beispiel wird immer derjenige Knoten ausgewählt, der dem Ziel am nächsten ist beziehungsweise der den größten Fortschritt in Bezug auf die Strecke vom Start- zum Zielknoten aufweist. Diese Strategie optimiert automatisch die Anzahl der Hops. Dementsprechend benötigen GRS und MFR im Vergleich zu den kürzesten Pfaden im Durchschnitt nur geringfügig mehr Hops, um den jeweiligen Zielknoten zu erreichen. Auch hinsichtlich der Pfadlänge in Metern sind beide Verfahren nur geringfügig schlechter als das Optimum. Bei einer Länge der kürzesten Pfade von etwa 2500 m benötigen beide Verfahren im Durchschnitt circa 2750 m, was einem zehnprozentigen Aufschlag entspricht. Das Compass-Verfahren benötigt zwar mehr Hops als GRS und MFR, aber die Pfadlänge in Metern unterscheidet sich nur geringfügig. Wesentlich stärker ist dieser Sachverhalt bei der Compass-Variante ausgeprägt, die auf dem planarisierten Netzwerkgraphen aufbaut. Befinden sich die Kommunikationspartner 14 Hops voneinander entfernt, so benötigt Compass-GG aufgrund der Planarisierung im Durchschnitt



**Abbildung 3.10:** Länge der gefundenen Pfade bei einer Funkreichweite von 300 m und einer Knotendichte von 25,37 Knoten/km.

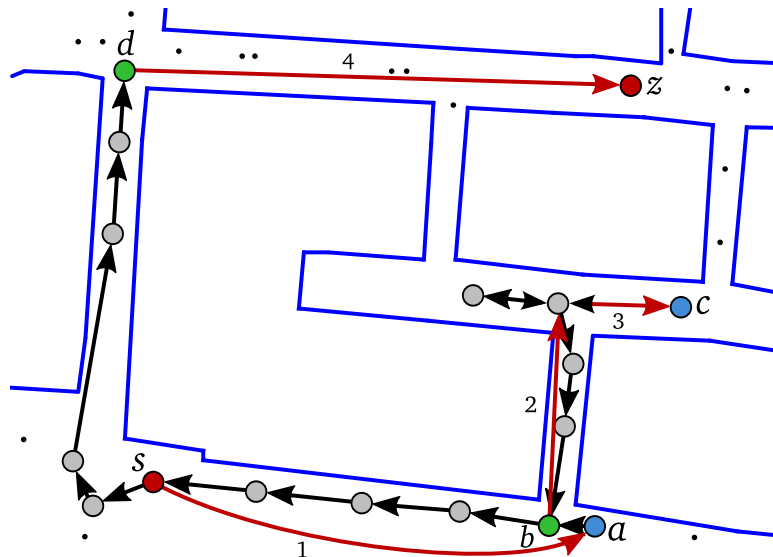
etwa 110 Hops, um einen Pfad zu finden. Auch die Pfadlängen in Metern sind bei diesen Verfahren am längsten, wenngleich der Abstand zu den anderen Heuristen hier wesentlich niedriger ausfällt. Da ein Knoten das Medium mit allen anderen Netzteilnehmern innerhalb der Funkreichweite teilt, ist die Pfadlänge in Hops bei einer konstanten Funkreichweite von wesentlich höherer Bedeutung als die Pfadlänge in Metern.

### 3.3.3 Greedy-Algorithmen mit Überbrückungsverfahren

Die im vorherigen Abschnitt präsentierten Ergebnisse zeigen, dass die Routingheuristen Compass, GRS und MFR für viele Kommunikationspartner in einem Innenstadtbereich mit Funkhindernissen keinen Pfad finden und somit für ein derartiges Einsatzgebiet ungeeignet sind. Wie in Abschnitt 2.1.4 beschrieben, können diese Heuristiken allerdings durch eine sogenannte Überbrückungsstrategie ergänzt werden. Immer wenn die Basisstrategie keinen geeigneten Weiterleiter findet, wird zum alternativen Routingverfahren gewechselt, das dann – ausgehend vom aktuellen Knoten – einen eigenen Pfad zum Zielknoten sucht. Im Allgemeinen übertragen die Überbrückungsstrategien während der Pfadsuche deutlich mehr Pakete als die Basisstrategien, sodass baldmöglichst auf das Basisverfahren zurückgewechselt werden sollte. Daher werden vor dem Start der Überbrückung sowohl die Position als auch der Identifikator des aktuellen Knotens im Paketkopf hinterlegt. Sobald ein Paket im Überbrückungsmodus empfangen wird, überprüft die Basisstrategie anhand der lokalen Informationen und des Paketkopfs, ob der aktuelle Knoten oder einer seiner Nachbarn eine bessere Bewertung erzielt als der Knoten, der die Überbrückung gestartet hat. Sollte dies der Fall sein, wird die alternative Pfadsuche abgebrochen und das Basisverfahren wieder aufgenommen. Der detaillierte Ablauf

wird in Abschnitt 3.1 erläutert und in Abbildung 3.2 in Form eines UML-Aktivitätsdiagramms visualisiert.

Hinter dieser Kombination steht die Idee, eine sehr effiziente und einfache Heuristik, die aber keine erfolgreiche Pfadsuche garantieren kann, mit einem zweiten, zuverlässigeren, aber weniger effizienten Verfahren zu ergänzen. Aus diesem Grund werden in dieser Arbeit nur Face-2 und ein AODV-ähnlicher Ansatz zur Überbrückung herangezogen. Um die erfolgreiche Pfadsuche des kombinierten Verfahrens zu garantieren, muss allerdings die Bewertungsmetrik des Basisverfahrens global gültig sein. Ansonsten kann während des alternativen Weiterleitens die Eignung des aktuellen Knotens und seiner Nachbarn nicht mit dem Startknoten der Überbrückung verglichen werden. Da Compass den Nachbarn  $n$  auswählt, der – ausgehend vom aktuellen Knoten  $v$  und dem Zielknoten  $z$  – den kleinsten Winkel  $\angle vzn$  formt, ist die Bewertung von  $n$  auch vom aktuellen Knoten abhängig und kann somit nicht mit dem Winkel aus der Sicht eines anderen Knotens verglichen werden. Von der Überbrückungsstrategie abgesehen kann daher bei Compass nicht sichergestellt werden, dass die Auswahl des nächsten Weiterleiters immer auch zu einem Fortschritt innerhalb des Bewertungssystems führt. Im Gegensatz dazu wird in der verwendeten Implementierung von GRS und MFR nach der Auswahl des nächsten Weiterleiters immer dessen Abstand zum Zielknoten mit demjenigen des aktuellen Knotens verglichen. Wenn das Weiterleiten nicht zu einer Reduktion des Abstands führt, wird die Pfadsuche abgebrochen beziehungsweise der Überbrückungsmodus aufgenommen. Aufgrund dieses Sachverhalts werden im Folgenden nur GRS und MFR mit den beiden Überbrückungsstrategien evaluiert.

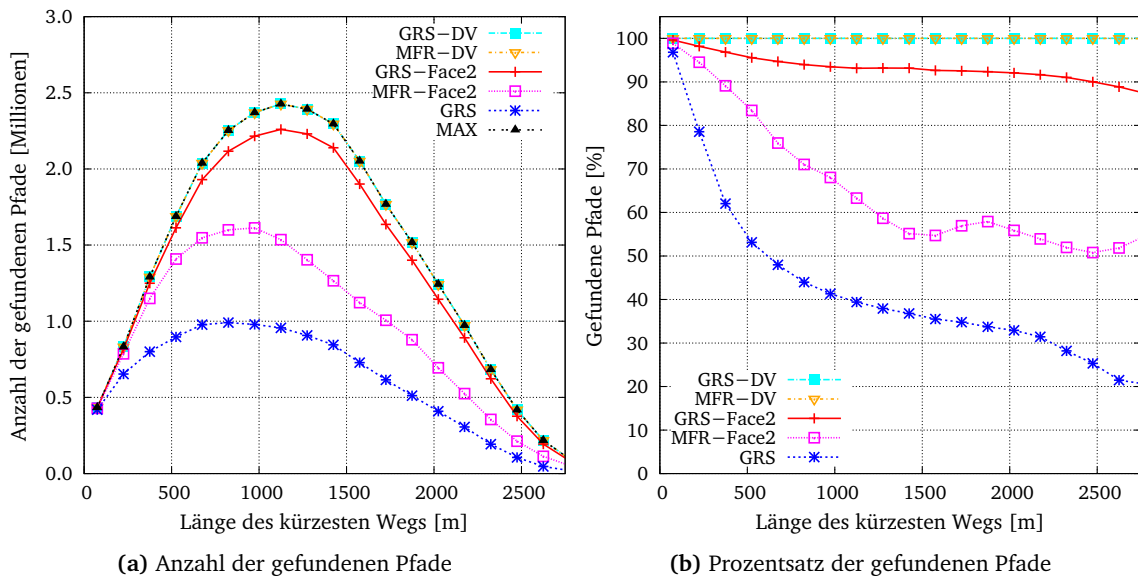


**Abbildung 3.11:** Die Pfadsuche von GRS mit Face-2 als Überbrückungsstrategie (rote Kanten: GRS, schwarze Kanten: Face-2, blaue Knoten: Wechsel von GRS auf Face-2, grüne Knoten: Wechsel von Face-2 auf GRS).



Abbildung 3.11 zeigt eine exemplarischen Pfadsuche von GRS mit Face-2 als Überbrückungsstrategie. Ausgehend vom Startknoten  $s$  wird der Knoten  $a$  als nächster Weiterleiter ausgewählt, weil dieser von allen Nachbarn räumlich am nächsten am Zielknoten  $z$  liegt. Da  $a$  keinen Nachbarn besitzt, der näher an  $z$  liegt als er selbst, wird in den Überbrückungsmodus gewechselt, und  $b$  wird als nächster Weiterleiter bestimmt. Auf Knoten  $b$  wechselt die Weiterleitungsstrategie wieder zurück auf GRS, weil Knoten  $b$  einen Nachbarn besitzt, der näher an  $z$  liegt als  $a$ . Nach zwei Hops befindet sich GRS auf dem Knoten  $c$  in einem lokalen Optimum, sodass erneut zu Face-2 gewechselt wird. Anschließend wird die Pfadsuche von Face-2 auf dem planarisierten Graphen bis zum Knoten  $d$  weitergeführt, von dem aus der Zielknoten  $z$  direkt erreicht werden kann.

Die Simulationsergebnisse von GRS und MFR mit den beiden Überbrückungsstrategien werden in Abbildung 3.12 dargestellt. Für jedes Verfahren ist links die absolute Anzahl der gefundenen Pfade eingezeichnet und rechts der resultierende Prozentsatz. Sowohl GRS-DV als auch MFR-DV finden für alle möglichen Kommunikationspartner einen Pfad. Die dabei verwendete topologiebasierte Überbrückungsstrategie ist an AODV angelehnt und sucht mit lokalem Fluten nach einem Knoten, der gemäß der Basisstrategie besser geeignet ist als der aktuelle. Hierbei wird in der verwendeten Implementierung mit einem Suchradius von 1 begonnen, der so lange schrittweise auf maximal 15 erhöht wird, bis ein geeigneter Knoten gefunden wurde. Je größer der maximale Suchradius ist, desto höher ist auch die Wahrscheinlichkeit einer erfolgreichen Überbrückung. Die in der Abbildung gezeigten Ergebnisse basieren, wie erwähnt, auf einem maximalen Suchradius von 15 Hops. Detaillierte Ergebnisse mit unterschiedlichen Suchradien werden in Abschnitt 3.5 präsentiert.

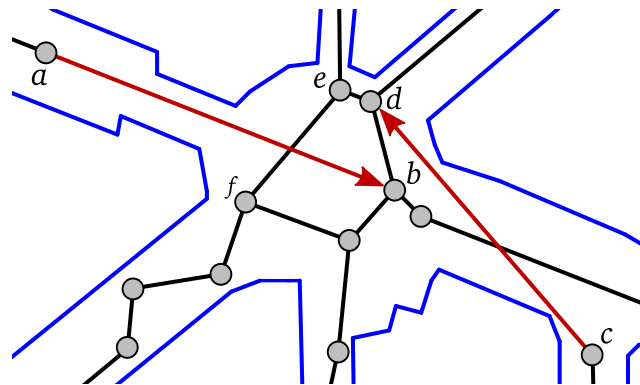


**Abbildung 3.12:** Ergebnisse der Pfadsuche bei einer Funkreichweite von 300 m und einer Knotendichte von 25,37 Knoten/km.



Analog zu den Ergebnissen ohne Überbrückung schneidet die Kombination von GRS mit Face-2 besser ab als die Variante mit MFR. GRS mit Face-2 findet beispielsweise bei einer Pfadlänge von etwa 1500 m circa 93 % aller Pfade, während die Kombination mit MFR nur 55 % findet. Auch wenn GRS mit Face-2 wesentlich bessere Resultate erzielt als ohne diese Überbrückungsstrategie, gibt es doch weiterhin eine signifikante Menge von Knotenpaaren, die nicht miteinander kommunizieren können. Die Implementierung von Face-2 wurde intensiv getestet und findet in einem Szenario ohne Funkhindernisse alle Pfade. Die Ursache für die schlechteren Ergebnisse in einem Gebiet mit Funkhindernissen liegt nicht am eigentlichen Verfahren, sondern an einer fehlerhaften Planarisierung des Netzwerkgraphen.

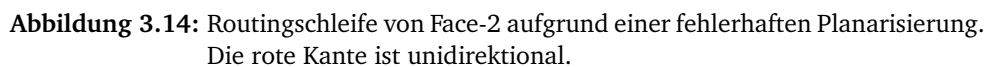
In einem Gabriel Graph sind zwei Knoten  $u$  und  $v$  nur dann miteinander durch eine Kante verbunden, wenn in dem Kreis mit der Strecke  $uv$  als Durchmesser kein anderer Knoten des Graphen liegt. Wenn jeder Knoten des Netzwerks durch Nachrichtenaustausch die Positionen der anderen Knoten innerhalb seiner Funkreichweite kennt, kann der planare Gabriel Graph verteilt berechnet werden, indem jeder Knoten nur die ausgehenden Kanten verwendet, die der Konstruktionsbedingung entsprechen. Daher kann die Planarisierung als eine Filteroperation auf der Nachbarschaftstabelle angesehen werden. Wie Abbildung 3.11 zeigt, werden bei der verteilten Planarisierung vor allem die langen Kanten ignoriert, was im Durchschnitt zu deutlich längeren Pfaden führt. Neben der Planarität besitzt der Gabriel Graph noch die wichtige Eigenschaft, dass alle Knoten einer Zusammenhangskomponente nach der Planarisierung auch weiterhin miteinander verbunden sind.



**Abbildung 3.13:** Der Gabriel Graph im Innenstadtszenario mit Funkhindernissen. Die rot markierten Kanten sind unidirektional und können bei Face-2 zu Routingschleifen führen.

Abbildung 3.13 zeigt einen Ausschnitt aus dem Simulationsgebiet mit einer zufälligen Knotenverteilung. Es sind nur die zum Gabriel Graph gehörigen Kanten eingezeichnet. Im Unterschied zu allen anderen Kanten sind diejenigen von  $a$  nach  $b$  sowie von  $c$  nach  $d$  unidirektional. Kein Nachbar von  $a$  befindet sich innerhalb des Kreises mit dem Durchmesser  $ab$ , sodass die Kante  $\overrightarrow{ab}$  aus der Sicht von  $a$  zum Gabriel Graph gehört. Da sich allerdings aus der Sicht von  $b$  mehrere andere Knoten innerhalb des Kreises befinden, wird die Kante  $\overrightarrow{ba}$  von Knoten  $b$  nicht

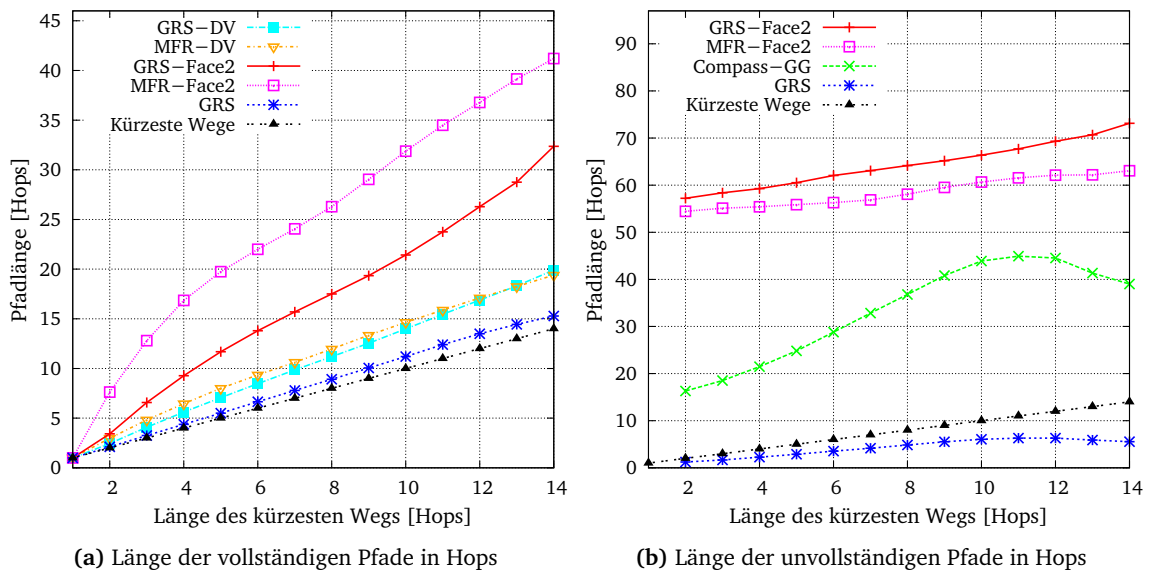
Durch den Austausch der Nachbarschaftstabellen kann in einigen Fällen die Planarisierung verbessert werden. Auf diese Weise können beispielsweise die Knoten  $e$  und  $f$  eine Überschneidung mit der Kante  $\overrightarrow{ab}$  vermeiden, indem sie ihre gemeinsame Kante ignorieren. Des Weiteren kann durch den Nachrichtenaustausch auch die Kante  $\overrightarrow{ab}$  in eine bidirektionale Kante umgewandelt werden. Allerdings ist es prinzipiell nicht möglich, die Überschneidungsproblematik in allen Fällen zufriedenstellend aufzulösen. Beispielsweise kann die Überschneidung der Kante  $\overrightarrow{cd}$  nur durch ignorieren einer der beiden beteiligten Kanten vermieden werden, was in beiden Fällen zu einer Partitionierung des Netzwerks führen kann. Sollte dies der Fall sein, könnte Face-2 keinen Pfad zwischen zwei Knoten finden, die zu unterschiedlichen Netzwerkpartitionen des Gabriel Graph gehören. Selbst wenn die beteiligten Knoten weiterhin über Umwege miteinander verbunden sein sollten, würde sich zumindest die Konnektivität des Netzwerks deutlich verschlechtern. Wie Abbildung 3.12 entnommen werden kann, benötigt Face-2 aufgrund der Planarisierung vergleichsweise viele Hops, um einen Pfad zu finden. Die verschlechterte Konnektivität würde dementsprechend die Pfadlänge nochmals signifikant steigern.



66

kleinsten Winkel  $\angle szn$  formt. In dem betrachteten Beispiel ist dies der Knoten  $a$ . Weil der Winkel  $\angle szn$  im Uhrzeigersinn berechnet kleiner ist als entgegen dem Uhrzeigersinn, startet Face-2 mit der Rotationsrichtung im Uhrzeigersinn. Sei  $v$  der jeweils aktuell betrachtete Knoten und  $u$  der direkte Vorgänger von  $v$ . Um einen Pfad entlang der Grenze des ersten Faces zu finden, wählt der Face-2-Algorithmus immer denjenigen Nachbar  $n$  aus, der im Uhrzeigersinn den kleinsten Winkel  $\angle uvn$  formt. Dieser Auswahlprozess würde normalerweise so lange fortgeführt werden, bis die zum ausgewählten Knoten führende Kante die Strecke  $sz$  schneidet. Da allerdings die Kante  $\overrightarrow{bc}$  in dem verteilt berechneten Gabriel Graph unidirektional ist, wird das Paket immer wieder um den nördlichen Häuserblock herum weitergeleitet und befindet sich also in einer Routingschleife.

Abbildung 3.15 zeigt auf der linken Seite die Länge der gefundenen Pfade. GRS und MFR benötigen mit der an AODV angelehnten Überbrückung nur wenige Hops mehr als der jeweils kürzeste Pfad. Für zwei Knoten, die sich auf dem kürzesten Pfad 10 Hops voneinander entfernt befinden, benötigt GRS-DV beispielsweise im Durchschnitt etwa 14 Hops. Die Varianten mit Face-2 als Überbrückungsstrategie schneiden deutlich schlechter ab. Im vorherigen Beispiel benötigte GRS-Face2 beispielsweise mit 21 Hops im Durchschnitt etwa 6 Hops mehr als GRS-DV. Dabei ist allerdings zu beachten, dass im Unterschied zu Face-2 die DV-Überbrückung den Pfad über einen Flutmechanismus findet und somit die Werte nicht unmittelbar vergleichbar sind. Die Eignung beider Verfahren wird detailliert in Abschnitt 3.5 erläutert.



**Abbildung 3.15:** Länge der vollständigen und der unvollständigen Pfade bei einer Funkreichweite von 300 m und einer Knotendichte von 25,37 Knoten/km.

Für einige Einsatzszenarien ist es ausreichend, wenn für mehr als 90 % aller potenziellen Kommunikationspartner ein Pfad gefunden werden kann. Insbesondere in sehr mobilen Szenarien sollte ohnehin davon ausgegangen werden, dass regelmäßig temporäre Kommunikati-

onsprobleme auftreten können. Von dieser Überlegung abgesehen, sind die von Face-2 gefundenen Pfade allerdings vergleichsweise lang. In Einsatzgebieten mit Funkhindernissen besteht zusätzlich das Risiko, dass die Pfadsuche in eine Endlosschleife gerät. Aus diesem Grund wird in der Simulation ein Überbrückungsvorgang nach 50 Hops abgebrochen.

Im Folgenden werden die Pfade von erfolglosen Suchvorgängen unabhängig von der Abbruchsursache unvollständig genannt. Auch wenn diese Pfade nicht zum jeweiligen Zielknoten führen, belasteten die zugehörigen Suchvorgänge dennoch das Netzwerk. Abbildung 3.15b zeigt daher die durchschnittliche Länge aller unvollständigen Pfade. Die unvollständigen Pfade der beiden Varianten mit Face-2 als Überbrückungsstrategie sind durchgehend mehr als 55 Hops lang, wobei die MFR-Variante etwas besser abschneidet. Dies liegt allerdings nur an dem Umstand, dass GRS-Face2 deutlich mehr Pfade findet und somit die verbleibenden Pfade im Durchschnitt eine höhere Komplexität aufweisen. Die Länge der unvollständigen Pfade ist bei Compass-GG ebenfalls sehr hoch, liegt aber noch deutlich unter dem Niveau der vollständigen Pfade. Da bei GRS der nächste Weiterleiter immer näher am Ziel liegen muss als der aktuelle Knoten, wird die Pfadsuche im Durchschnitt nach einer kleineren Anzahl Hops abgebrochen als die beiden Kommunikationspartner auf dem kürzesten Pfad voneinander entfernt sind.

### 3.3.4 Greedy-Algorithmen mit Kreuzungserkennung

Face-2 kann als Überbrückungsstrategie die Anzahl der gefunden Pfade signifikant erhöhen. Allerdings wird das Netzwerk aufgrund der langen Pfade und des Risikos von Endlosschleifen stark belastet. Die Varianten mit der an AODV angelehnten Überbrückungsstrategie finden zwar deutlich kürzere Pfade und können Endlosschleifen ausschließen, aber der dabei eingesetzte Flutmechanismus kann das Netzwerk ebenfalls stark belasteten. Aufgrund dieses Sachverhalts wird in diesem Abschnitt untersucht, ob die Routingheuristiken mit Hilfe einer Kreuzungserkennung derart verbessert werden können, dass die *teuren* Überbrückungsstrategien signifikant seltener eingesetzt werden müssen.

Die Straßentopologie kann als Graph aufgefasst werden, indem die Kreuzungen als Knoten und die Straßen als Kanten interpretiert werden. Abgesehen von Ausnahmen wie beispielsweise Überführungen oder Brücken ist der resultierende Graph überwiegend planar. Die Idee von *Greedy Perimeter Coordinator Routing* (GPCR) [103, 101] besteht darin, diesen planaren Graphen als übergeordnetes Hilfsmittel bei der Pfadsuche zu verwenden. Zu Beginn der Pfadsuche wird das Paket mit *Restricted Greedy Routing* (RGR) in Richtung des Zielknotens weitergeleitet. Die Besonderheit von RGR im Vergleich zu GRS besteht darin, dass diejenigen Knoten, die sich auf einer Straßenkreuzung befinden, bevorzugt ausgewählt werden. Sollte während der Pfadsuche beim Knoten  $x$  ein lokales Optimum auftreten, wechselt RGR in den Überbrückungsmodus, welcher das Paket gemäß der Rechte-Hand-Regel von Straßenkreuzung zu Straßenkreuzung in Richtung des Zielknotens weitergeleitet. In diesem Überbrückungsmodus wird ebenfalls RGR eingesetzt, um einen Pfad entlang der ausgewählten Straßen zu finden. Sobald ein Knoten gefunden wird, der näher am Zielknoten liegt als derjenige im lokalen Op-

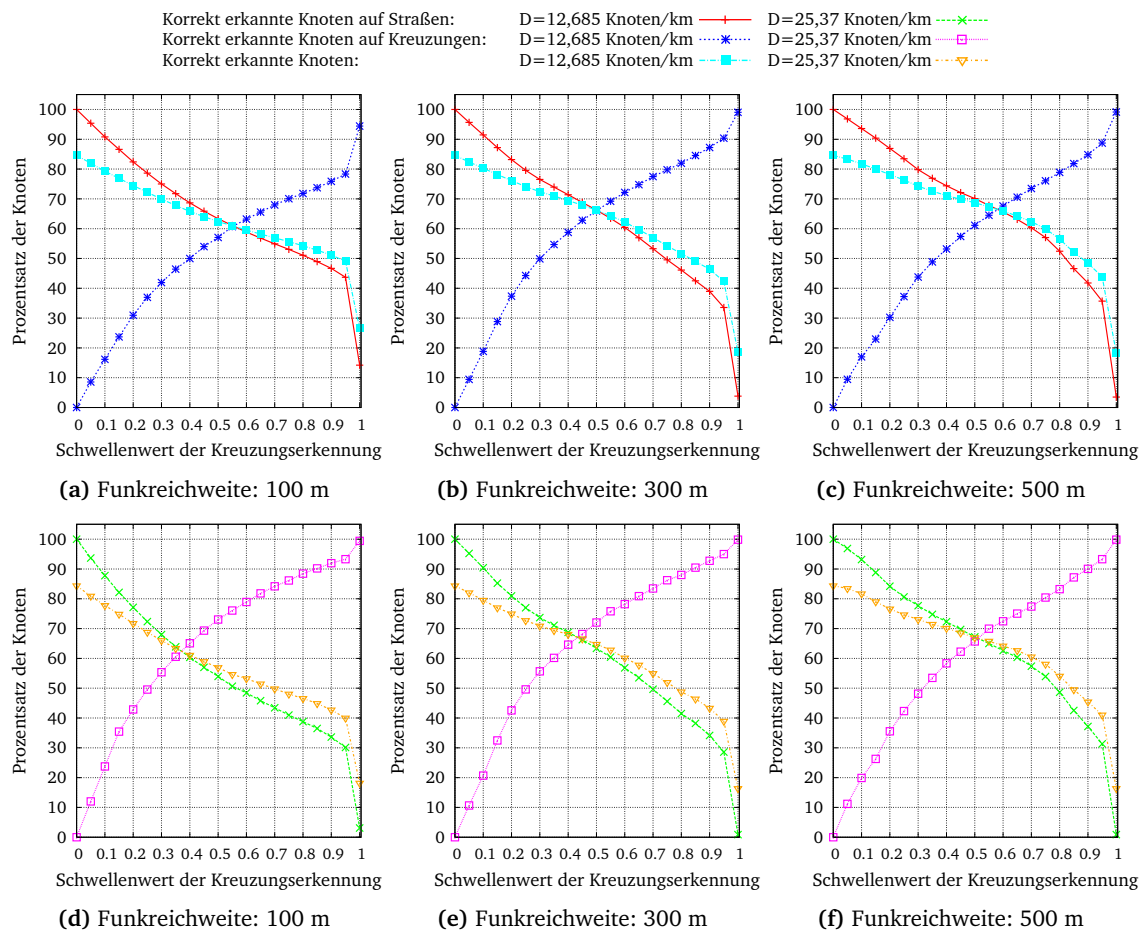
timum, wird die Überbrückung beendet und das Weiterleiten direkt mit RGR in Richtung des Zielknotens fortgesetzt.

Zusätzlich zum eigenen Identifikator und der aktuellen Position übertragen bei RGR die Knoten in regelmäßigen Abständen auch ein Statusbit, das angibt, ob sich der jeweilige Knoten auf einer Kreuzung befindet. Um dies festzustellen, schlagen die Autoren zwei dezentrale Verfahren vor. Das erste setzt eine 2-Hop-Nachbarschaftstabelle voraus, die durch zusätzlichen Datenaustausch zwischen den jeweils benachbarten Knoten aufgebaut werden muss. Besitzt ein Knoten  $c$  zwei Nachbarn, deren geografischer Abstand untereinander kleiner ist als die Funkreichweite, die sich aber nicht gegenseitig in ihren Nachbarschaftslisten aufführen, so befindet sich  $c$  vermutlich auf einer Kreuzung, und die beiden Nachbarn sind durch ein Gebäude voneinander getrennt. Für das zweite Verfahren ist die Kenntnis der 1-Hop-Nachbarschaft ausreichend, die ohnehin für GRS und RGR benötigt wird. Um zu beurteilen, ob sich ein Knoten auf einer Kreuzung befindet, wird anhand der  $x$ - und  $y$ -Koordinaten seiner Nachbarn der Korrelationskoeffizient  $p_{xy} \in [0, 1]$  berechnet (siehe Formel 2.1). Liegt der Wert  $p_{xy}$  nahe bei eins, sind die Positionen der Nachbarn linear voneinander abhängig, sodass sich der Knoten wahrscheinlich auf einer Straße befindet. Anhand eines geeigneten Schwellenwerts  $s$  kann für jeden Knoten der vermutete Aufenthaltsort festgelegt werden, indem das Statusbit bei allen Knoten mit  $p_{xy} < s$  auf „Kreuzung“ und bei allen anderen Knoten auf „Straße“ gesetzt wird.

In den verwendeten Simulationsszenarien wird jeder Schnittpunkt von drei oder mehr Straßenabschnitten als Kreuzung interpretiert. Jeder Knoten, der nicht weiter als acht Meter von einem dieser Schnittpunkte entfernt ist, wird als Kreuzungsknoten betrachtet. Bei allen anderen Knoten wird davon ausgegangen, dass sie sich auf einer Straße befinden. Diese Aufenthaltsorte können vom Simulator mit den Ergebnissen der beiden Verfahren zur Kreuzungserkennung verglichen werden. Da allerdings der Aufbau einer 2-Hop-Nachbarschaftstabelle das Netzwerk sehr stark belastet, wird im Folgenden nur der Korrelationskoeffizient für die Kreuzungserkennung herangezogen. Hierbei ist insbesondere von Interesse, welcher Schwellenwert für eine Innenstadtumgebung am besten geeignet ist.

Abbildung 3.16 zeigt die Ergebnisse der Kreuzungserkennung mit Hilfe des Korrelationskoeffizienten bei unterschiedlichen Knotendichten und Funkreichweiten. Auf der  $x$ -Achse ist der jeweils verwendete Schwellenwert und auf der  $y$ -Achse der Prozentsatz der korrekt erkannten Knoten aufgetragen. Bei einer Funkreichweite von 100 m und einem Schwellenwert von 0,9 werden in den Szenarien mit der niedrigen Knotendichte etwa 78 % aller Knoten auf Kreuzungen richtig erkannt. Durch eine Erhöhung der Funkreichweite kann dieser Wert auf circa 88 % gesteigert werden. In den Szenarien mit hoher Knotendichte werden bei allen untersuchten Funkreichweiten etwa 90 % aller Knoten auf Kreuzungen richtig erkannt. Das Problem an diesem Verfahren besteht darin, dass mit einer steigenden Erkennungsquote von Knoten auf Kreuzungen gleichzeitig der Prozentsatz der korrekt erkannten Knoten auf Straßen sinkt. Bei einer Funkreichweite von 100 m und einem Schwellenwert von weiterhin 0,9 werden in den Szenarien mit niedriger Knotendichte nur rund 44 % aller Knoten auf Straßen korrekt erkannt. Dementsprechend wird bei den übrigen 56 % fälschlicherweise davon ausgegangen, dass sie

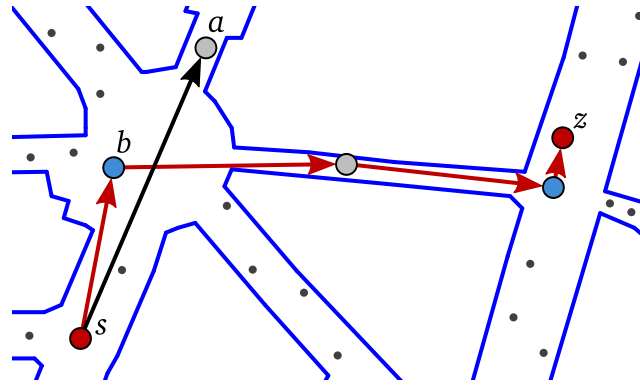
sich auf einer Kreuzung befinden. In den Szenarien mit hoher Knotendichte ist der Prozentsatz der korrekt erkannten Knoten auf Straßen mit 30 % sogar noch niedriger. Da in den untersuchten Szenarien unabhängig von der Knotendichte etwa 15,5 % aller Knoten auf einer Kreuzung und die verbleibenden 84,5 % auf einer Straße liegen, werden insgesamt am meisten Knoten korrekt eingestuft, wenn ein Schwellenwert von 0,0 verwendet wird. In diesem Fall wird unabhängig vom eigentlichen Korrelationskoeffizienten davon ausgegangen, dass sich alle Knoten auf Straßen befinden, was wiederum die Kreuzungserkennung ad absurdum führt. Eine vergleichbare Problematik ergibt sich auch beim Einsatz der Kreuzungserkennung mit Hilfe der 2-Hop-Nachbarschaftstabelle.



**Abbildung 3.16:** Ergebnisse der Kreuzungserkennung anhand des Korrelationskoeffizienten bei unterschiedlichen Knotendichten und Funkreichweiten.

In allen untersuchten Konfigurationen sind die Ergebnisse der Kreuzungserkennung bei Weitem nicht ausreichend, um im Überbrückungsmodus Face-2 auf den ermittelten Straßengraphen anzuwenden. Da RGR die Knoten auf den Kreuzungen nur bevorzugt, aber keine weitere Rückschlüsse auf die Straßentopologie benötigt, kann möglicherweise selbst eine nicht voll-

ständig funktionierende Kreuzungserkennung zu einer Verbesserung gegenüber GRS führen. Abbildung 3.17 zeigt einen Kartenausschnitt, in dem die Pfadsuche von GRS mit und ohne Kreuzungserkennung dargestellt wird. Ausgehend von dem Startknoten  $s$  wählt GRS den Knoten  $a$  aus, weil dieser den kleinsten Abstand zum Zielknoten  $z$  aufweist. In diesem Beispiel gerät GRS dann allerdings in ein lokales Optimum. Im Unterschied zu GRS wählt RGR zu Beginn der Pfadsuche den Knoten  $b$  als nächsten Weiterleiter aus, obwohl dieser weiter von  $z$  entfernt ist als  $a$ . Während GRS in diesem Beispiel auf den Einsatz einer Überbrückungsstrategie angewiesen ist, kann RGR durch die Bevorzugung von Knoten auf Kreuzungen selbstständig einen Pfad zum Zielknoten  $z$  finden. Wie die Abbildung zeigt, haben Knoten auf Kreuzungen generell den Vorteil, dass sie die Knoten auf den angrenzenden Straßen bei der Weiterleiterauswahl berücksichtigen können.



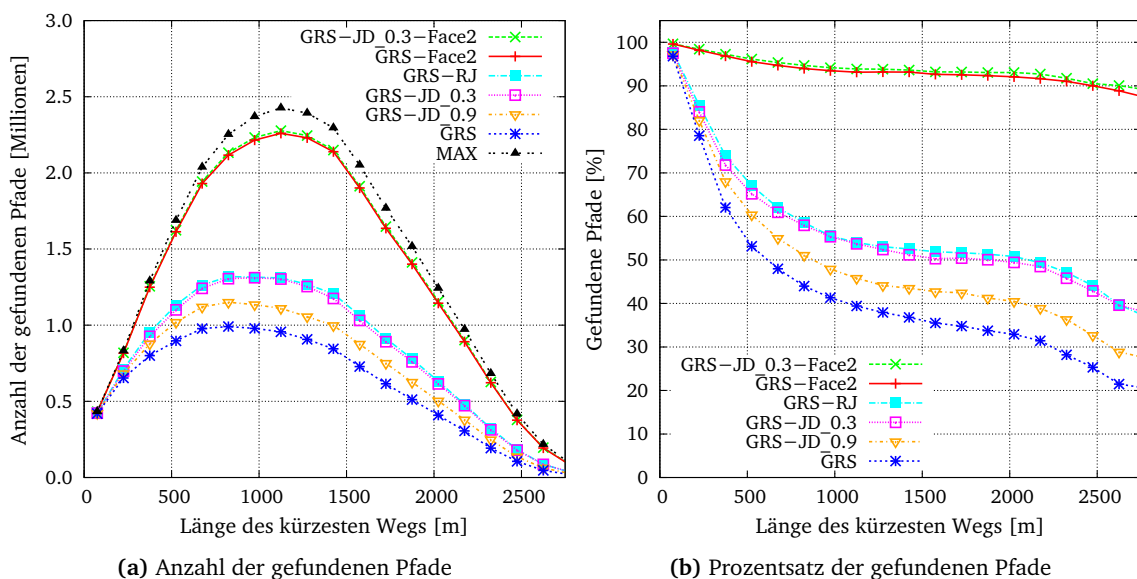
**Abbildung 3.17:** Die Pfadsuche von GRS mit und ohne Kreuzungserkennung (blaue Knoten: Knoten auf Kreuzungen, rote Kanten: GRS mit Kreuzungserkennung, schwarze Kante: GRS ohne Kreuzungserkennung).

Wie die Priorisierung von Knoten auf einer Kreuzung bei der Weiterleiterauswahl im Detail realisiert werden soll, ist in [103] nur vage beschrieben. In der hier verwendeten Implementierung wird für jeden Nachbarknoten anhand der Metrik des jeweiligen Routingverfahrens der Abstand zum Zielknoten errechnet. Der Knoten mit dem kleinsten Abstand wird als nächster Weiterleiter ausgewählt. Sollte dieser jedoch gemäß der verwendeten Metrik einen größeren Abstand zum Zielknoten aufweisen als der aktuelle Knoten, wird die Weiterleitung abgebrochen und es wird (insofern vorhanden) in den Überbrückungsmodus gewechselt. Diese Vorgehensweise ist aber nur gültig, wenn die Eignung eines Knotens mit einer eindimensionalen Metrik ausgedrückt werden kann. Bei GRS wird der geometrische Abstand zum Zielknoten als Metrik verwendet. Durch eine Priorisierung von Knoten auf Kreuzungen wird gewissermaßen eine zweite Dimension geschaffen. Eine Möglichkeit, beide Bewertungskriterien mit einem skalaren Zahlenwert auszudrücken, besteht darin, bei allen Knoten auf Straßen einen fixen Zusatzwert zu dem realen Abstand zum Zielknoten hinzuzurechnen. Da die Pfadsuche allerdings abgebrochen wird, wenn kein Knoten mit einem kleineren Abstand verfügbar ist, wäre es bei dieser kombinierten Metrik nicht mehr möglich, von Knoten auf Kreuzungen zu solchen



auf Straßen zu wechseln. Des Weiteren gibt es Sonderfälle zu berücksichtigen; beispielsweise muss der Zielknoten unabhängig von seinem Aufenthaltsort immer den kleinstmöglichen Abstand aufweisen.

Aus diesen Gründen verwendet der Simulator bei Verfahren mit Kreuzungserkennung einen modifizierten Auswahlmechanismus. Bei der eigentlichen Berechnung des Abstands gemäß der verwendeten Metrik wird die Kreuzungserkennung nicht berücksichtigt. Stattdessen wird mit Hilfe der Abbruchbedingung schon bei der Bewertung der Nachbarknoten getestet, ob deren Auswahl zu einem Abbruch führen würde. Von den Knoten, bei denen dies nicht der Fall ist, wird derjenige Knoten ausgewählt, der den kleinsten Abstand zum Zielknoten aufweist. Dabei werden die Knoten auf Kreuzungen unabhängig vom konkreten Abstand bevorzugt. Diese abgeänderte Vorgehensweise ist notwendig, um die folgende Problematik zu vermeiden: Der Knoten  $a$  hat die beiden Nachbarn  $b$  und  $c$ , wobei sich der Knoten  $b$  auf einer Kreuzung befindet und der Knoten  $c$  auf einer Straße. Würde nun die Bevorzugung der Knoten auf Kreuzungen mit dem bisherigen Auswahlmechanismus kombiniert werden, so würde der Knoten  $b$  ausgewählt werden. Wenn allerdings der Knoten  $b$  einen größeren Abstand zum Zielknoten hat als der Knoten  $a$ , würde gemäß der Abbruchbedingung die Pfadsuche mit der Basisstrategie eingestellt werden, obwohl Knoten  $c$  möglicherweise ein geeigneter Weiterleiter wäre.

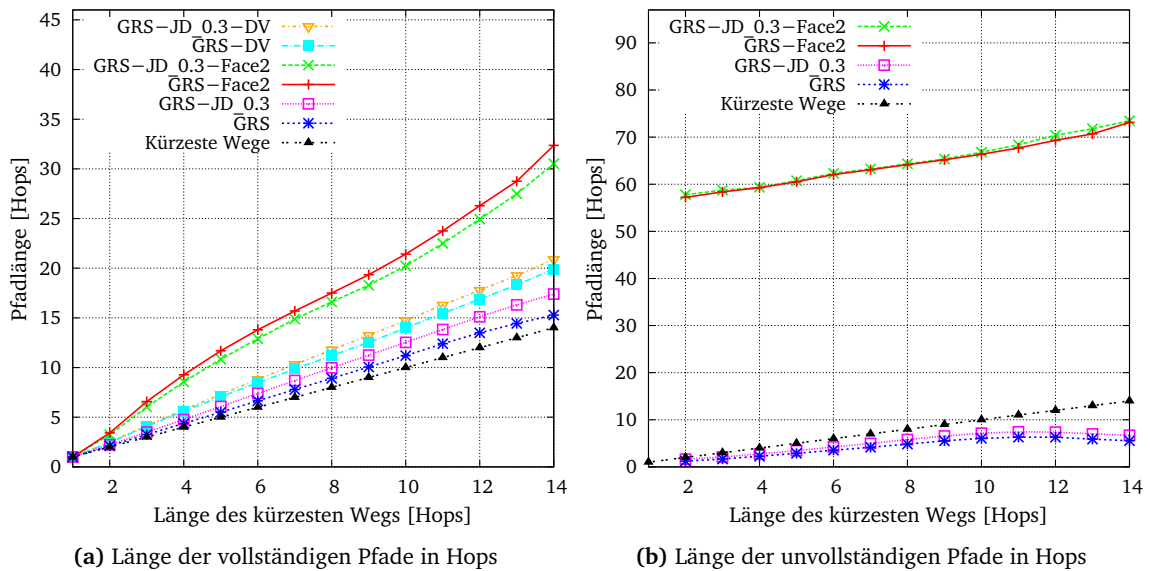


**Abbildung 3.18:** Ergebnisse der Pfadsuche bei einer Funkreichweite von 300 m und einer Knotendichte von 25,37 Knoten/km.

Abbildung 3.18 zeigt die Simulationsergebnisse von GRS mit Kreuzungserkennung. Die Kreuzungserkennung per Korrelationskoeffizient wird mit JD (Junction Detection) abgekürzt. Der hinter JD stehende Wert gibt den jeweils verwendeten Schwellenwert an. Auf der linken Seite der Abbildung wird, wie zuvor, die Anzahl der gefunden Pfade dargestellt, auf der rechten Seite die resultierenden Prozentsätze. Für etwa 40 % der Kommunikationspartner, die auf dem



kürzesten Pfad circa 2000 m voneinander entfernt sind, kann GRS-JD bei einem Schwellenwert von 0,9 einen Pfad finden. Diese Variante schneidet somit etwa acht Prozentpunkte besser ab als GRS ohne Kreuzungserkennung. Um die Grafik übersichtlich zu gestalten, wurden die Ergebnisse von GRS mit der Kreuzungserkennung anhand der 2-Hop-Nachbarschaftstabelle nicht eingezeichnet. Sie liegen aber in etwa auf dem Niveau von GRS-JD\_0.9. Unter dem Namen GRS-RJ ist in der Grafik auch eine Variante von GRS zu sehen, die Zugriff auf die Straßentopologie hat und somit von jedem Knoten weiß, ob er sich auf einer Kreuzung befindet oder nicht. Diese primär für Vergleichszwecke verwendete Implementierung findet für rund 50 % aller Kommunikationspartner, die auf dem kürzesten Pfad circa 2000 m voneinander entfernt sind, einen Pfad. Wird bei der Kreuzungserkennung anhand des Korrelationskoeffizienten ein Schwellenwert von 0,3 verwendet, erzielt GRS-JD nahezu die gleichen Ergebnisse wie GRS-RJ, sodass im Folgenden ausschließlich die Variante mit dem Korrelationskoeffizienten weiter betrachtet wird.



**Abbildung 3.19:** Länge der vollständigen und der unvollständigen Pfade bei einer Funkreichweite von 300 m und einer Knotendichte von 25,37 Knoten/km.

Wie GRS kann auch GRS-JD mit Face-2 als Überbrückungsstrategie eingesetzt werden. Doch obwohl die Basisstrategie mit Kreuzungserkennung signifikant mehr Pfade findet als diejenige ohne, sind die Ergebnisse beider Varianten mit Überbrückung nahezu identisch. Zumindest ergibt sich der Vorteil, dass wesentlich seltener auf die Überbrückungsstrategie zurückgegriffen werden muss. Abbildung 3.19 zeigt auf der linken Seite die Länge der gefundenen Pfade und auf der rechten Seite die Länge der unvollständigen Pfade. Durch den selteneren Einsatz von Face-2 benötigt GRS-JD\_0.3-Face2 im Durchschnitt etwa ein bis zwei Hops weniger als GRS-Face2. Allerdings verbleibt die Pfadlänge weiterhin auf einem sehr hohen Niveau. Diese vergleichsweise geringe Verbesserung begründet sich dadurch, dass der Einsatz der Kreuzungser-

kennung die von der Basisstrategie gefundenen Pfade verlängert, wodurch teilweise der seltenere Einsatz der Überbrückung ausgeglichen wird. Da die an AODV angelehnte Überbrückung durch einen lokalen Flutmechanismus einen optimalen Überbrückungspfad ermittelt, wirkt sich die Pfadverlängerung der Basisstrategie unmittelbar negativ auf das Gesamtergebnis aus. Auch bei den unvollständigen Pfaden erhöht die Kreuzungserkennung die durchschnittliche Länge geringfügig. Allerdings ist die Länge dieser Pfade bei Verfahren mit Face-2 als Überbrückungsstrategie primär von der Limitierung der Länge eines Überbrückungsversuchs abhängig, sodass in dieser Disziplin beide Varianten nahezu identische Ergebnisse erzielen.

### 3.4 Greedy Routing with Abstract Neighbour Table

Durch den Einsatz einer Kreuzungserkennung kann die Anzahl der von GRS gefundenen Pfade signifikant erhöht werden. Wird allerdings GRS mit Face-2 als Überbrückungsstrategie ergänzt, findet die Variante mit Kreuzungserkennung nur geringfügig mehr Pfade als das Pendant ohne Kreuzungserkennung. Da die Priorisierung von Knoten auf Kreuzungen unabhängig von deren Fortschritt zum Ziel die durchschnittliche Pfadlänge der Basisstrategie erhöht, kann trotz des selteneren Einsatzes der Überbrückungsstrategie die durchschnittliche Pfadlänge nur unwesentlich reduziert werden. Die Strategie GRS mit einer Kreuzungserkennung zu verbessern ist somit nur teilweise gelungen. Daher wird in den folgenden Abschnitten ein neu entwickeltes Verfahren vorgestellt.

#### 3.4.1 Theoretische Vergleichsverfahren

Nachdem ein Knoten aktiviert wurde, kennt dieser nur den eigenen Identifikator und gegebenenfalls die eigene Position. Die meisten Routingverfahren sind deshalb darauf angewiesen, mittels Nachrichtenaustausch zusätzliche Informationen über die Topologie des Netzwerks zu erlangen. Durch einen verteilten Wettbewerb können manche Verfahren (vgl. Abschnitt 2.1.2) auch ohne explizite Kenntnis der benachbarten Knoten auskommen und den nächsten Weiterleiter dennoch anhand der GRS-Heuristik auswählen. Bei dieser Vorgehensweise können allerdings nur Informationen genutzt werden, die den benachbarten Knoten schon vorliegen. Die Kreuzungserkennung benötigt beispielsweise zur Klassifizierung des Aufenthaltsorts die Positionen der direkten Nachbarn, sodass sie nicht mit einem Wettbewerbsverfahren kombiniert werden kann.

Topologiebasierte Verfahren wie AODV nutzen einen Flutmechanismus, um einen Pfad zum Zielknoten zu finden. Dabei werden ebenfalls Informationen über die Topologie der Umgebung gesammelt und dezentral in den Routingtabellen der beteiligten Knoten gespeichert. Falls sich die Knoten gar nicht und nur sehr langsam bewegen, kann es sinnvoll sein, die Pfade proaktiv für alle möglichen Kommunikationspartner zu ermitteln und in den Routingtabellen zu hinterlegen. Je schneller sich allerdings die Knoten bewegen, desto häufiger müssen die gespeicherten Informationen erneuert werden. Des Weiteren ist die Aktualität der Tabelleneinträge

auch von der Entfernung zum jeweiligen Knoten abhängig. Beispielsweise ist die Wahrscheinlichkeit, dass ein langer, explizit aufgebauter Pfad durch Knotenbewegungen seine Gültigkeit verliert, höher als bei einem Pfad mit kürzerer Länge. Dementsprechend muss in Abhängigkeit von der zu erwartenden Gültigkeitsdauer der ausgetauschten Informationen ein geeignetes Aktualisierungsintervall gewählt werden. Doch je häufiger die Informationen aktualisiert werden müssen, desto mehr Daten werden pro Zeiteinheit zwischen den Knoten ausgetauscht. Gleichzeitig belastet die Aktualisierung von weiter entfernten Informationen alle Knoten auf dem jeweiligen Pfad. Deshalb ist es vorteilhaft, wenn das Routingverfahren mit möglichst lokalen Informationen auskommt.

Die meisten bisher besprochenen positionsbasierten Verfahren verwenden ausschließlich Informationen ihrer direkten Nachbarn. Dabei schickt jeder Knoten in regelmäßigen Abständen seinen Identifikator, seine Position und gegebenenfalls den Status der Kreuzungserkennung an alle Nachbarn innerhalb seiner Funkreichweite. Abgesehen von dem möglicherweise notwendigen Dienst zur Bestimmung der Zielposition sind diese Verfahren aufgrund der ausschließlichen Nutzung der 1-Hop-Topologie sehr gut für hoch mobile Umgebungen geeignet. Da allerdings bei allen nicht gefunden Pfaden ohnehin auf eine aufwendige Überbrückungsstrategie zurückgegriffen werden muss, ist es eine valide Vorgehensweise, die Zuverlässigkeit der Basisverfahren durch zusätzliche Informationen zu verbessern. Aus diesem Grund werden im Folgenden zwei Verfahren vorgestellt, anhand derer Ergebnisse abgeschätzt werden kann, wie stark GRS von Zusatzwissen profitieren kann.

Alle Knoten innerhalb eines  $x$ -Hop-Radius um den Knoten  $c$  herum werden im Folgenden mit  $NB_{c,x} = \{ n \mid kwl(n, c) \leq x \}$  bezeichnet, wobei die Funktion  $kwl(v_1, v_2)$  die Länge des kürzesten Wegs zwischen den Knoten  $v_1$  und  $v_2$  in Hops ermittelt. Das für Vergleichszwecke entwickelte Routingverfahren *GRS- $x$ Hop* setzt voraus, dass jeder Netzwerknoten  $c \in V$  von allen Nachbarn  $n \in NB_{c,x}$  den Identifikator, die Position und den jeweiligen Abstand kennt. Im Detail bedeutet dies, dass jeder Knoten  $c \in V$  eine  $NB_{c,x-1}$  umfassende Nachbarschaftstabelle aufbaut und diese an seine direkten Nachbarn überträgt. Bei  $x = 1$  besteht  $NB_{c,0}$  für jeden Knoten  $c \in V$  nur aus dem jeweiligen Knoten, sodass jeder Knoten ausschließlich die eigenen Daten an seine direkten Nachbarn überträgt, welche hierdurch eine 1-Hop-Nachbarschaftstabelle aufbauen können. Diese Vorgehensweise entspricht exakt der Strategie der bisher besprochenen positionsbasierten Routingverfahren. Wie die bisherigen Verfahren wählt auch GRS- $x$ Hop denjenigen direkten Nachbarn als nächsten Weiterleiter aus, der gemäß der verwendeten Metrik den kleinsten Abstand zum Zielknoten aufweist. Die dabei verwendete Metrik berechnet das Abstandsmaß eines direkten Nachbarn, indem ausgehend sowohl von dessen Position als auch von denen seiner Nachbarn der Abstand zum Zielknoten berechnet und der kleinste hierbei gefundene Abstand ausgewählt wird. Da von dem aktuellen Knoten  $c$  aus die Knoten  $NB_{n,x-1}$  nur über den direkten Nachbarn  $n \in NB_{c,1}$  erreicht werden können, wird bei der Abstandsrechnung der Nachbarn von  $n$  ein Aufschlag verwendet, der sicherstellt, dass direkt erreichbare Knoten eine bessere Bewertung erzielen als Knoten, die nur über Umwege zu erreichen sind. In den folgenden Simulationen wird ein Aufschlagfaktor von 1,05 pro zu überwindendem Hop

verwendet. Die detaillierte Abstandsberechnung vom Knoten  $n \in NB_{c,1}$  zum Zielknoten  $d$  wird in Algorithmus 3.1 beschrieben.

---

**Algorithmus 3.1 :** Die Berechnung des Abstands von Knoten  $n \in NB_{c,1}$  zum Zielknoten  $d$  gemäß der GRS-xHop-Metrik.

---

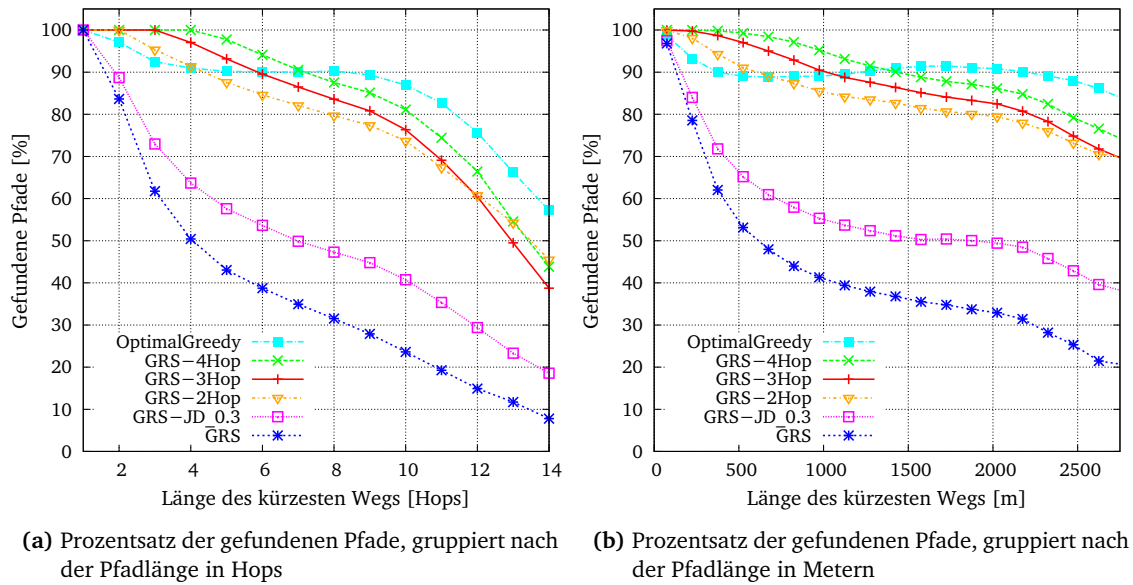
```
if  $n = d$  then
  return  $-\infty$ ;
 $minAbstand := d(n, d)$ ;
for  $m \in NB_{n,x-1}$  do
  if  $m = d$  then
     $abstand := -x + kwl(n, m)$ ;
  else
     $abstand := d(m, d) \cdot kwl(n, m) \cdot AufschlagFaktor$ ;
  if  $abstand < minAbstand$  then
     $minAbstand := abstand$ ;
return  $minAbstand$ ;
```

---

Auch bei GRS-xHop wird nach der Auswahl eines Weiterleiters getestet, ob dieser gemäß der verwendeten Metrik näher am Ziel liegt als der aktuelle Knoten. Doch selbst wenn dies der Fall sein sollte, ist aufgrund der in Algorithmus 3.1 dargestellten Berechnungsweise nicht sichergestellt, dass der nächste Knoten auch den euklidischen Abstand zum Ziel verringert. Durch diesen Umstand kann GRS-xHop mit  $x > 1$  Pfadverläufe zum Ziel entdecken, die von GRS aufgrund der verwendeten Abbruchbedingung prinzipiell nicht gefunden werden können. Dementsprechend sind alle denkbaren Auswahlverfahren, die erzwingen, dass der euklidische Abstand pro Hop reduziert wird, in Bezug auf die Menge möglicher Pfadverläufe, die ein Verfahren finden kann, weniger mächtig als GRS-xHop mit  $x > 1$ . Um den Einfluss dieses theoretischen Aspekts in den folgenden Simulationsergebnissen besser bewerten zu können, wird mit *OptimalGreedy* ein zweites Vergleichsverfahren verwendet.

*OptimalGreedy* umfasst die Idee, aus der Menge aller möglichen Pfadverläufe, die pro Hop den euklidischen Abstand verringern, den jeweils kürzesten Pfad zum Zielknoten auszuwählen. Dieses Verfahren stellt somit das Optimum dar, welches von einem dezentralen Auswahlmechanismus mit der beschriebenen Abbruchbedingung theoretisch erreicht werden kann. Da bisher kein dezentrales Verfahren bekannt ist, das anhand der 1-Hop-Nachbarschaftstabelle dieses Optimum bestimmen kann, verwendet der Simulator einen modifizierten *Dijkstra-Algorithmus* [45]. Dabei werden auf der Suche nach dem kürzesten Pfad jeweils nur diejenigen ausgehenden Kanten berücksichtigt, die den euklidischen Abstand zum Ziel nicht vergrößern.

Abbildung 3.20 zeigt die Ergebnisse der Pfadsuche von *OptimalGreedy* und GRS-xHop mit  $x = 2, 3, 4$  innerhalb des Stadt szenarios. Die Ergebnisse sind auf der linken Seite nach der Pfadlänge in Hops und auf der rechten Seite nach der Pfadlänge in Metern gruppiert. GRS-xHop



**Abbildung 3.20:** Ergebnisse der Pfadsuche von OptimalGreedy und GRS- $x$ Hop mit  $x = 2, 3, 4$  bei einer Funkreichweite von 300 m und einer Knotendichte von 25,37 Knoten/km.

findet für alle Kommunikationspartner, die sich auf dem kürzesten Pfad  $x$  Hops voneinander entfernt befinden, einen Pfad. Bei einem Abstand zwischen fünf und zehn Hops gestalten sich die Kurvenverläufe aller drei Varianten etwa parallel, wobei immer diejenige Variante besser abschneidet, die mehr Informationen bei der Wegewahl berücksichtigt. Im weiteren Kurvenverlauf fällt das Ergebnis von GRS-3Hop hinter die beiden anderen Varianten zurück. Dieser Umstand entspricht allerdings nur einer unbedeutenden statistischen Schwankung, verursacht von der vergleichsweise geringen Anzahl von Kommunikationspartnern, die sich auf dem kürzesten Pfad elf oder mehr Hops voneinander entfernt befinden. Werden die Ergebnisse nach dem euklidischen Abstand entlang des kürzesten Pfades gruppiert, sind die Ergebnisse deutlich besser verteilt, sodass die zuvor beschriebenen Schwankungen bei längeren Pfaden nur noch abgemildert auftreten.

Für zwei Kommunikationspartner, die sich auf dem kürzesten Pfad acht Hops voneinander entfernt befinden, besteht eine Chance von etwa 32 %, einen Pfad mit Hilfe von GRS zu finden. Der Einsatz einer Kreuzungserkennung erhöht diese Wahrscheinlichkeit auf circa 48 %. Mit Hilfe einer 2-Hop-Nachbarschaftstabelle kann GRS-2Hop die Wahrscheinlichkeit für zwei derartige Kommunikationspartner, einen Pfad zu finden, auf 80 % erhöhen. Für  $x = 3$  und  $x = 4$  findet GRS- $x$ Hop mit einer Wahrscheinlichkeit von 84 % beziehungsweise 88 % einen Pfad. Allerdings sind diese Zuwächse im Vergleich zu den 48 Prozentpunkten, die GRS-2Hop gegenüber GRS zulegt, recht niedrig. Auch wenn in diesem Abschnitt der Algorithmus zum Aufbau einer Multi-Hop-Nachbarschaftstabelle außer Acht gelassen wird, zeigen die Ergebnisse doch,

dass GRS-xHop für  $x = 2$  das beste Verhältnis zwischen der erreichten Ergebnisverbesserung und dem zu erwartenden Kommunikationsmehraufwand aufweist.

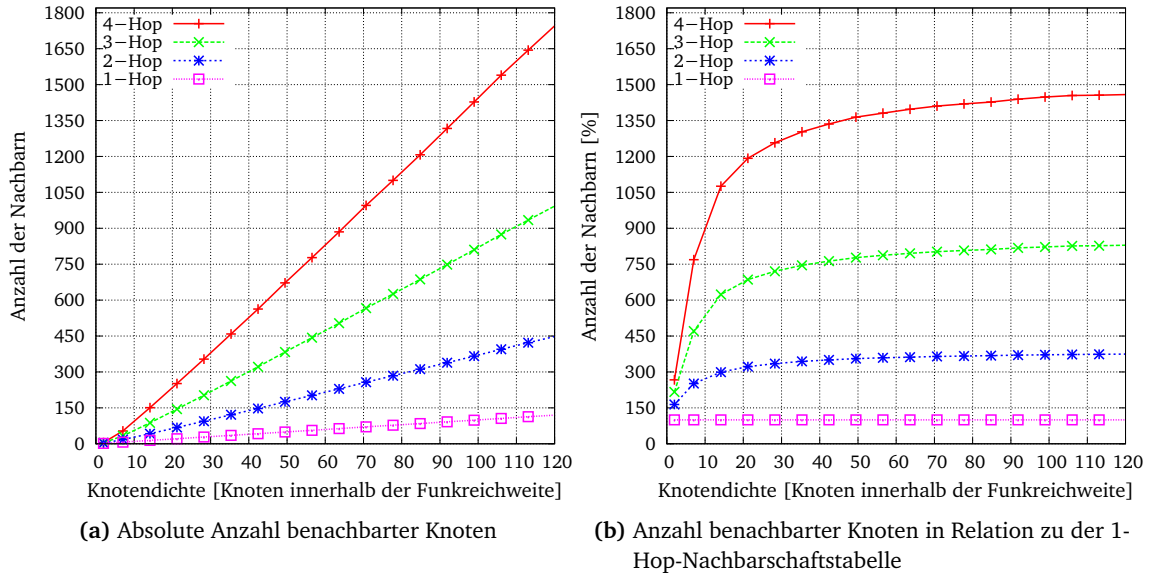
Mit Hilfe von OptimalGreedy kann das theoretische Optimum evaluiert werden, das ein Routingverfahren erreichen kann, welches pro Hop den euklidischen Abstand zum Ziel reduziert. Da es im verwendeten Stadtszenario Knoten gibt, die trotz einer räumlichen Nähe nur über Umwege miteinander verbunden werden können, findet OptimalGreedy im Unterschied zu den GRS-xHop-Varianten auch bei einem Kommunikationsabstand von nur zwei Hops nicht für alle Kommunikationspartner einen Pfad. Befinden sich allerdings die Knoten entlang des kürzesten Wegs mehr als 7 Hops beziehungsweise etwa 1450 m voneinander entfernt, so findet OptimalGreedy sogar mehr Pfade als GRS-4Hop. Damit ist die Anzahl der Kommunikationspartner, für die ein Pfad existiert, der pro Hop den euklidischen Abstand zum Ziel verringert, wesentlich größer als die Anzahl der Kommunikationspartner, für die GRS mit Kreuzungserkennung einen Pfad findet. Im Unterschied zu den GRS-xHop-Varianten ist allerdings unbekannt, welche Informationen ein Algorithmus benötigen würde, um die von OptimalGreedy gefunden Pfade *dezentral* zu bestimmen. Aus diesem Grund konzentriert sich die folgende Analyse auf GRS-xHop.

### 3.4.2 Abstract Neighbour Table

Die Ergebnisse von GRS-xHop zeigen, dass die Anwendung der Weiterleiteraushwahl von GRS auf eine Multi-Hop-Nachbarschaftstabelle für einen hohen Prozentsatz von möglichen Kommunikationspartnern einen Pfad finden kann. Abbildung 3.21 zeigt die zu erwartende Anzahl von Einträgen in einer Multi-Hop-Nachbarschaftstabelle in Abhängigkeit von der durchschnittlichen Anzahl an Knoten innerhalb der verwendeten Funkreichweite. Bei einer Reichweite von 500 m entsprechen beispielsweise 40 Knoten innerhalb der Funkreichweite circa 51 Knoten pro km<sup>2</sup>. Da die Knoten am Rand des Simulationsgebiets aufgrund der räumlichen Begrenzung weniger Nachbarn besitzen als solche in der Mitte, werden bei der Simulation nur Knoten berücksichtigt, die mindestens das Vierfache der Funkreichweite von den Grenzen des Simulationsgebiets entfernt sind.

Unabhängig davon, wie der Aufbau der Nachbarschaftstabellen im Detail verläuft, ist die Anzahl der Einträge ein Indiz für den hierfür notwendigen Kommunikationsaufwand. Im Fall einer 1-Hop-Nachbarschaftstabelle besteht diese offensichtlich immer aus exakt so vielen Einträgen, wie sich Knoten innerhalb der jeweiligen Funkreichweite befinden. Wenn der Identifikator eines Knotens 32 Bit benötigt und der Breiten- sowie der Längengrad jeweils durch eine 32-Bit-Gleitkommazahl repräsentiert werden, benötigt jeder Eintrag in der Nachbarschaftstabelle 12 Byte. Bei 50 Knoten innerhalb der Funkreichweite wächst die Nachbarschaftstabelle bei einem Radius von zwei Hops auf circa 180 Knoten beziehungsweise 2160 Byte und bei einem Radius von drei Hops auf circa 340 Knoten beziehungsweise 4080 Byte.

Auf der rechten Seite der Abbildung sind die Ergebnisse relativ zur Größe der 1-Hop-Nachbarschaftstabelle eingezeichnet, die für den Einsatz von nahezu allen positionsbasierten Verfahren notwendig ist. Bei höheren Knotendichten stehen die dargestellten Größen in einem



**Abbildung 3.21:** Die durchschnittliche Anzahl von Einträgen in einer Multi-Hop-Nachbarschaftstabelle in einem Simulationsgebiet ohne Funkhinder-nisse.

annähernd konstanten Verhältnis zueinander. Während eine 2-Hop-Nachbarschaftstabelle etwa 3-mal so viel Speicherplatz benötigt wie die 1-Hop-Variante, benötigt eine Nachbarschaftstabelle mit einem 4-Hop-Radius schon etwa das 14-Fache.

Wie im vorherigen Abschnitt beschrieben, benötigt GRS-xHop für die Weiterleiterauswahl von allen direkten Nachbarn die  $x-1$ -Hop-Nachbarschaftstabellen. Wenn sich im Durchschnitt etwa 50 Knoten innerhalb der Funkreichweite befinden, benötigt der Tabellenaustausch aus der Sicht eines Knotens bei GRS-2Hop circa  $50 \cdot 50 \cdot 12 \text{ Byte} = 30000 \text{ Byte}$ . Bei GRS-3Hop und GRS-4Hop steigt der jeweils notwendige Datenaustausch auf  $50 \cdot 180 \cdot 12 \text{ Byte} = 108000 \text{ Byte}$  beziehungsweise  $50 \cdot 340 \cdot 12 \text{ Byte} = 204000 \text{ Byte}$  an. Da GRS-xHop mit  $x = 3$  und  $x = 4$  nur geringfügig bessere Resultate erzielt als GRS-2Hop, bietet die letztgenannte Variante den besten Kompromiss zwischen dem zusätzlich notwendigen Datenaustausch und der jeweiligen Verbesserung bei der Pfadsuche. Zusätzlich altern die ausgetauschten Nachbarschaftsinformationen in mobilen Szenarien mit jedem zusätzlichen Hop schneller und müssten deshalb in immer kürzeren Zeitintervallen aktualisiert werden.

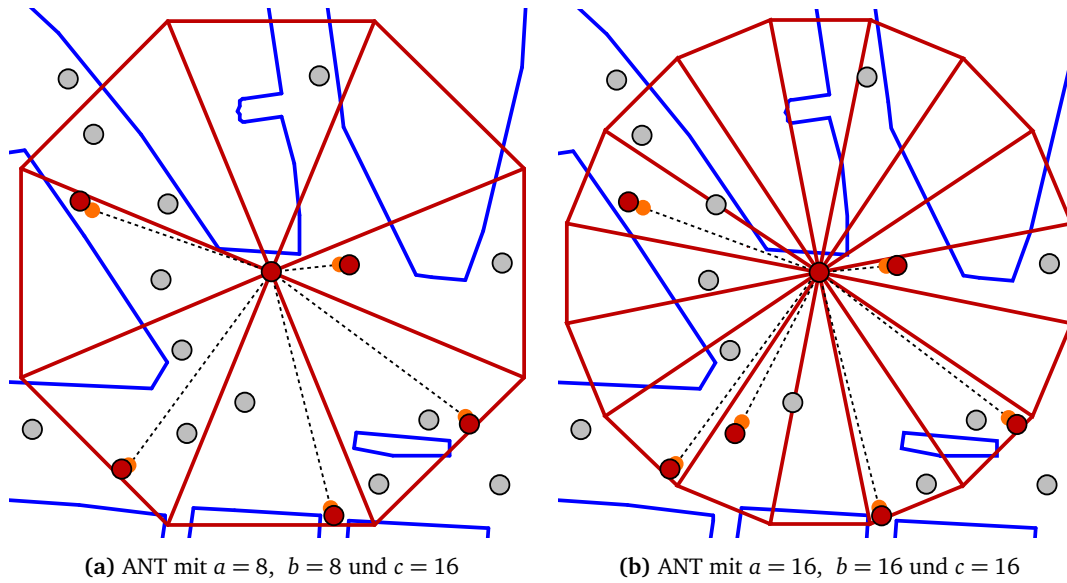
Auch wenn GRS-2Hop aus den dargestellten Gründen den besten Kompromiss bietet, ist der zusätzliche proaktive Kommunikationsaufwand dennoch enorm und vor allem quadratisch von der Knotendichte abhängig. Gleichzeitig ist bei einer hinreichend hohen Knotendichte der Informationsgewinn durch einen zusätzlichen Eintrag in der Nachbarschaftstabelle nur noch äußerst gering. So haben beispielsweise zwei Knoten mit einem sehr kleinen Abstand zueinander gemäß der in Algorithmus 3.1 beschriebenen Metrik auch einen ähnlich großen Abstand zum Ziel, sodass die Auflistung beider Nachbarn wenig effizient ist. Die *Abstract Neighbour*



*Table* (ANT) reduziert deshalb den proaktiven Datenaustausch auf ein Minimum, indem die Anzahl der aufgelisteten Nachbarn limitiert wird. Wie der Name der Datenstruktur bereits andeutet, wird anstelle der gesamten Nachbarschaftstabelle nur noch eine kompakte Zusammenfassung übertragen. Dafür wird die Ebene um den betrachteten Knoten  $v$  in  $a$  gleich große Bereiche unterteilt, die gegen den Uhrzeigersinn von 0 bis  $(a - 1)$  durchnummeriert werden. Für jeden Bereich  $n \in 0, \dots, a - 1$  wird derjenige Knoten  $r_n$  als Repräsentant ausgewählt, der den größten Abstand zu  $v$  aufweist. Die Repräsentanten werden in der Reihenfolge ihrer Nummerierung in der ANT gespeichert. Damit die Anzahl der Einträge immer konstant ist und somit auf eine explizite Nummerierung verzichtet werden kann, werden die Bereiche ohne Knoten durch einen leeren Eintrag repräsentiert. Da abgesehen von der eindeutigen Identifizierung des Zielknotens die Identifikatoren der Repräsentanten nicht zwingend notwendig sind, werden diese nicht in der ANT gespeichert. Um den Speicherbedarf weiter zu reduzieren, werden die Positionen der Repräsentanten relativ zu der Position von  $v$  als Vektor hinterlegt. Jeder dieser Vektoren kann durch seine Länge und den Winkel zwischen der Grenze des jeweiligen Bereichs  $n$  und der Strecke  $vr_n$  beschrieben werden. Zur weiteren Reduktion des Speicherbedarfs werden die Wertebereiche der Länge und des Winkels jeweils quantisiert. Hierfür wird jeder Bereich in  $b$  Unterbereiche aufgeteilt, die von 0 bis  $(b - 1)$  durchnummeriert werden. Der Winkel wird dann gespeichert, indem die Nummer des Untergebiets, dessen Grenze am nächsten zum Repräsentanten liegt, hinterlegt wird. Die Abstände zwischen  $v$  und den Repräsentanten sind aufgrund der maximalen Funkreichweite  $F$  begrenzt, sodass der entsprechende Wertebereich zwischen 0 und  $F$  wiederum in  $(c - 1)$  gleich große Stücke unterteilt wird. Die Länge des zu hinterlegenden Vektors wird durch  $len = \min(c - 1, \text{floor}(\frac{d(v, r_n)}{F/(c-1)} + 1))$  bestimmt, wobei die Funktion *floor* die größtmögliche ganze Zahl zurückgibt, die nicht größer ist als der Parameter. Auf diese Weise wird sichergestellt, dass die hinterlegte Länge nicht größer ist als der reale Abstand zwischen  $v$  und dem entsprechenden Repräsentanten. Zur Codierung eines leeren Eintrags wird die Länge 0 verwendet. Unter der Annahme, dass auch für die leeren Einträge der gesamte Speicherplatz reserviert wird, kann der Speicherbedarf einer ANT mit  $s = \frac{a}{8}(\log_2 b + \log_2 c)[\text{Byte}]$  berechnet werden.

Abbildung 3.22 zeigt für zwei Konfigurationsbeispiele die räumliche Aufteilung der ANT. Die ausgewählten Repräsentanten sind in Rot und die in der ANT hinterlegten Positionsinformationen in Orange eingezeichnet. Die dargestellten ANT's unterteilen die Ebene in 8 beziehungsweise 16 Gebiete. Des Weiteren wird der Winkel durch 8 beziehungsweise 16 Untergebiete und die Länge in beiden Fällen durch 16 Intervalle codiert. Die Speicherung dieser ANT's benötigt 7 Byte beziehungsweise 16 Byte. Ob ein verschicktes Paket erfolgreich von einem Knoten empfangen werden kann, ist in realen Umgebungen nicht allein vom räumlichen Abstand der beteiligten Knoten anhängig. Deshalb wird sowohl in der dargestellten ANT als auch in den folgenden Simulationen anstatt der jeweiligen Funkreichweite ein wesentlich höherer Wert als  $F$  verwendet. In dem dargestellten Beispiel wird für  $F$  die zweifache Funkreichweite verwendet. Dieser Umstand und die oben beschriebene Berechnungsweise der gespeicherten Länge  $s$  führen dazu, dass die gespeicherten Winkel meist exakter sind als die gespeicherten Längen.





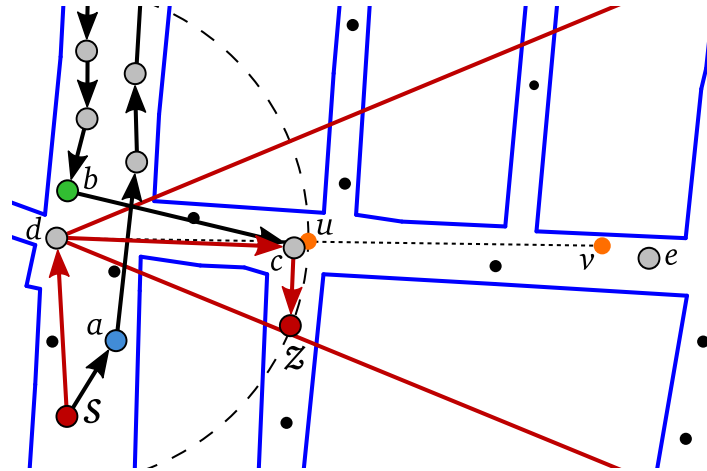
**Abbildung 3.22:** Zwei Varianten der Abstract Neighbour Table (ANT). Die ausgewählten Repräsentanten sind in Rot und die gespeicherten Positionen in Orange eingezeichnet.

### 3.4.3 Die Metrik von Greedy Routing with Abstract Neighbour Table

In diesem Abschnitt wird das Routingverfahren *Greedy Routing with Abstract Neighbour Table* (GRANT) [135, 143] vorgestellt. Die Idee dieses Verfahrens besteht darin, dass benachbarte Knoten anstelle vollständiger 1-Hop-Nachbarschaftstabellen ANTs austauschen und diese für die Weiterleiterauswahl nutzen. Im Gegensatz zu GRS-xHop wird dadurch der Kommunikationsmehraufwand von der Knotendichte entkoppelt und auf eine konstante Größe reduziert. Aufgrund des niedrigeren Informationsgehalts einer ANT muss allerdings die Bewertungsmetrik angepasst werden.

Abbildung 3.23 zeigt einen exemplarischen Ausschnitt aus dem Simulationsgebiet. Die Pfadsuche von GRS-Face2 ist in der Abbildung mit schwarzen Pfeilen eingezeichnet. Ausgehend vom Knoten  $s$  gerät GRS auf dem Knoten  $a$  in ein lokales Optimum, sodass zur Überbrückungsstrategie gewechselt wird. Auf der Suche nach einem geeigneten Weiterleiter besucht Face-2 alle Knoten der im Norden liegenden Sackgasse und findet anschließend den Knoten  $b$ , dessen Nachbar  $c$  näher an  $s$  liegt als der Knoten  $a$ . Aus diesem Grund wird in Knoten  $b$  zurück zur Basisstrategie gewechselt, die dann in zwei Hops den Zielknoten erreicht. Bei GRS-2Hop kennt  $s$  von allen Nachbarn wiederum deren Nachbarn und weiß deshalb, dass der Nachbar  $c$  von Knoten  $d$  näher an  $s$  liegt als  $a$ . Dementsprechend wählt GRS-2Hop  $d$  gleich als Weiterleiter aus und findet über den Knoten  $c$  einen Pfad zu  $z$ .

Im Unterschied hierzu kennt bei GRANT jeder Knoten nur die von den Nachbarn ausgewählten Repräsentanten. In dem dargestellten Beispiel ist Knoten  $s$  somit nur die ungefähre Position  $v$  des Repräsentanten  $e$  bekannt. Da der Knoten  $a$  näher am Ziel liegt als die codierte Position  $v$ ,



**Abbildung 3.23:** Die Pfadsuche von Knoten  $s$  zu Knoten  $z$  mit Hilfe von GRS-Face2 (schwarze Kanten) und GRANT (rote Kanten). Die in der ANT von  $d$  hinterlegte Position von  $e$  mit und ohne Längenlimitierung ist als orangener Punkt eingezeichnet.

würde  $a$  eine bessere Bewertung erhalten als  $d$ . Aus der ANT des Knotens  $d$  ist nicht ersichtlich, ob sich zwischen  $d$  und  $e$  noch andere Knoten befinden. In den durchgeführten Simulationsstudien zeigte sich, dass die durchschnittliche Anzahl der gefundenen Pfade signifikant erhöht werden kann, wenn die encodierte Länge des Vektors vom aktuell betrachteten Knoten  $v$  in Richtung des Repräsentanten auf den verbleibenden Abstand zum Zielknoten begrenzt wird. Der resultierende Punkt ist in der Abbildung als Punkt  $u$  eingezeichnet. Da  $u$  näher an  $z$  liegt als  $a$ , wählt GRANT  $d$  als Weiterleiter aus und findet den in Rot eingezeichneten Pfad zum Zielknoten.

Die Abstandsberechnung von Knoten  $n \in NB_{c,1}$  zum Zielknoten  $d$  inklusive der Längenlimitierung ist in Algorithmus 3.2 zu sehen. Während der Simulationsstudie wurden verschiedene Varianten der Längenlimitierung getestet. Unter anderem wird in einer Version die Länge nur limitiert, wenn der Repräsentant in der Richtung des Ziels liegt. Da die Variante, in der die Limitierung immer angewendet wird, mit circa sechs Prozentpunkten Vorsprung die besten Resultate erzielt, wird im Folgenden ausschließlich diese Version verwendet. In der dargestellten Abstandsberechnung wird des Weiteren auch ein Aufschlagfaktor eingesetzt, der gewährleistet, dass in den meisten Fällen ein Knoten  $v_1$  mit  $d(v_1, d) < d(v_2, d)$  eine bessere Bewertung erzielt als  $v_2$ , der  $v_1$  als Repräsentant in seiner ANT verwendet. Als Aufschlagfaktor wird in den folgenden Simulationen 1,05 verwendet.

Auch wenn GRANT im Durchschnitt mit der Längenlimitierung deutlich mehr Pfade findet als zuvor, kann diese Strategie in einigen Fällen zu unnötigen Umwegen oder gar zu erfolglosen Pfadsuchen führen. Abbildung 3.24 zeigt ein derartiges Beispiel. Ausgehend von dem Startknoten  $s$  wird unter anderem die Eignung von Knoten  $a$  mit der von Knoten  $d$  verglichen. Knoten  $a$  speichert in seiner ANT die Positionen der Knoten  $b$  und  $c$ . Da die Speicherung

---

**Algorithmus 3.2 :** Die Berechnung des Abstands von Knoten  $n \in NB_{c,1}$  zum Zielknoten  $d$  gemäß der GRANT-Metrik.

---

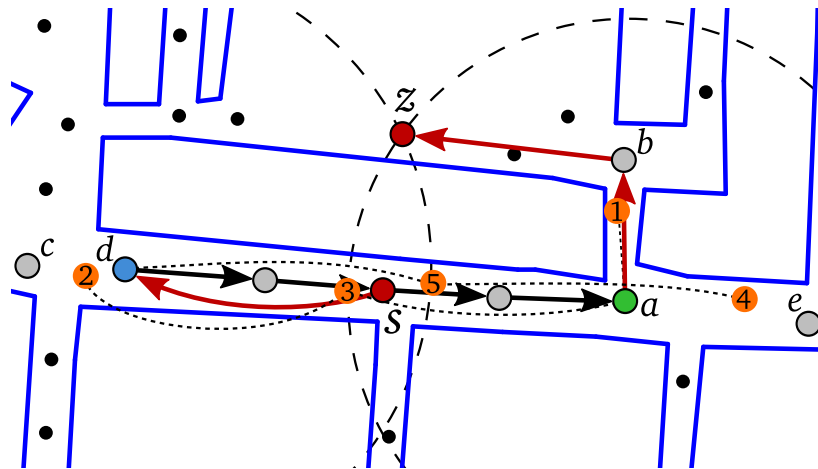
```

if  $n = d$  then
  return  $-\infty$ ;
 $minAbstand := d(n, d)$ ;
 $maxLänge := minAbstand$ ;
for  $h \in [0, a - 1]$  do
  if  $ANT[h].codierteLänge > 0$  then
     $winkel := decodiereWinkel(ANT[h].codierterWinkel)$ ;
     $länge := decodiereLänge(ANT[h].codierteLänge)$ ;
    if  $maxLänge < länge$  then
       $länge := maxLänge$ ;
     $r_h := berechneRepräsentantenPosition(n, winkel, länge)$ ;
     $abstand := d(r_h, d) \cdot AufschlagFaktor$ ;
    if  $abstand < minAbstand$  then
       $minAbstand := abstand$ ;
return  $minAbstand$ ;

```

---

verlustbehaftet erfolgt, sind die real hinterlegten Positionen separat als orangene Punkte mit den Nummern 1 und 2 eingezeichnet. Aufgrund der Längenlimitierung wird die Länge des gespeicherten Vektors von  $a$  zu 2 auf die Länge  $d(a, z)$  limitiert. Die resultierende Position ist in der Abbildung mit 3 beschriftet. Der Abstand von 3 zu  $z$  ist kleiner als der von 1, sodass der Knoten  $a$  die Bewertung  $d(3, z)$  erzielt. Der zweite von  $s$  berücksichtigte Knoten ist  $d$ , der unter anderem die Position von  $e$  in seiner ANT speichert. Die real hinterlegte Position ist als orangener Punkt mit der Nummer 4 visualisiert. Da auch in diesem Fall die Längenlimitierung angewendet wird, ist die bei der Bewertung berücksichtigte Position als 5 eingezeichnet. GRANT wählt den Knoten  $d$  als ersten Weiterleiter aus, weil dieser mit  $d(5, z)$  eine bessere Bewertung erzielt als der Knoten  $a$ . Auf dem Knoten  $d$  gerät GRANT in ein lokales Optimum und wechselt daher auf die in Schwarz eingezeichnete Überbrückungsstrategie. Der dabei verwendete Face-2-Algorithmus erreicht nach mehreren Zwischenstationen den Knoten  $a$ . Da der Nachbar  $b$  eine bessere Bewertung erzielt als der Knoten, bei dem die Überbrückung begonnen wurde, wechselt Face-2 auf dem Knoten  $a$  wieder zurück in den Basismodus. In dem dargestellten Beispiel hätte GRANT ohne die Längenlimitierung direkt einen Pfad zum Zielknoten gefunden und wäre nicht auf die Hilfe von Face-2 angewiesen gewesen. Doch trotz derartiger Fälle erhöht die Längenlimitierung im Durchschnitt die Anzahl der gefundenen Pfade signifikant und wird deshalb in allen Simulationen eingesetzt.



**Abbildung 3.24:** Die Wegewahl von GRANT (rote Kanten) gerät aufgrund der Längenlimitierung in ein lokales Optimum und kann nur mit Hilfe von Face-2 (schwarze Kanten) einen Pfad von Knoten  $s$  zu Knoten  $z$  finden.

### 3.4.4 Simulationsergebnisse

Für wie viel Prozent der möglichen Kommunikationspartner GRANT einen Pfad finden kann, hängt unter anderem von den verwendeten Konfigurationsparametern ab. Tabelle 3.2 zeigt die Simulationsergebnisse für alle GRANT-Varianten mit den Parameterkombinationen  $a = 4/8/16/32$ ,  $b = 0/4/8$  und  $c = 8/16/32$ . Während die ersten beiden Spalten alle Kombinationen der Parameter  $a$  und  $b$  auflisten, enthalten die letzten drei Spalten die jeweiligen Simulationsergebnisse für  $c = 8/16/32$ . Bei der Berechnung der Distanzschritte (DS) innerhalb der Simulation wurde die doppelte Funkreichweite als maximal codierbare Länge  $F$  verwendet.

**Tabelle 3.2:** Die Simulationsergebnisse von GRANT mit unterschiedlichen Konfigurationsparametern (Funkreichweite: 300 m, Knotendichte: 25,37 Knoten/km).

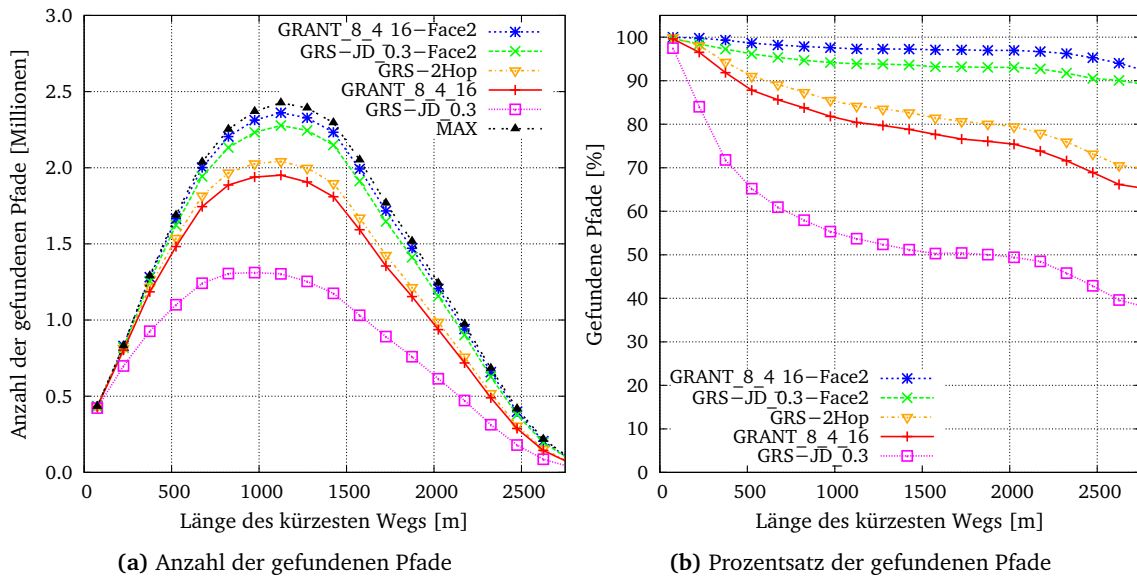
| Anzahl der Gebiete | Anzahl der Winkelschritte | Gefundene Pfade und benötigter Speicherplatz bei |           |                |                 |         |           |
|--------------------|---------------------------|--|-----------|----------------|-----------------|---------|-----------|
|                    |                           | 8 DS   |           | 16 DS          |                 | 32 DS   |           |
| 4                  | 0                         | 69,87 %  | (12 Bit)  | 70,70 %        | (16 Bit)        | 70,87 % | (20 Bit)  |
|                    | 4                         | 75,09 %  | (20 Bit)  | 75,77 %        | (24 Bit)        | 76,26 % | (28 Bit)  |
|                    | 8                         | 78,59 %  | (24 Bit)  | 79,35 %        | (28 Bit)        | 79,64 % | (32 Bit)  |
| 8                  | 0                         | 76,08 %  | (24 Bit)  | 76,84 %        | (32 Bit)        | 77,11 % | (40 Bit)  |
|                    | 4                         | 80,21 %  | (40 Bit)  | <b>81,05 %</b> | <b>(48 Bit)</b> | 81,43 % | (56 Bit)  |
|                    | 8                         | 80,69 %  | (48 Bit)  | 81.33 %        | (56 Bit)        | 81.58 % | (64 Bit)  |
| 16                 | 0                         | 79,37 %  | (48 Bit)  | 80,41 %        | (64 Bit)        | 80,75 % | (80 Bit)  |
|                    | 4                         | 81,01 %  | (80 Bit)  | 81,79 %        | (96 Bit)        | 82,16 % | (112 Bit) |
|                    | 8                         | 81,47 %  | (96 Bit)  | 82,49 %        | (112 Bit)       | 82.82 % | (128 Bit) |
| 32                 | 0                         | 80,04 %  | (96 Bit)  | 80,71 %        | (128 Bit)       | 81,03 % | (160 Bit) |
|                    | 4                         | 81,65 %  | (160 Bit) | 82,53 %        | (192 Bit)       | 82,85 % | (224 Bit) |
|                    | 8                         | 81,71 %  | (192 Bit) | 82,55 %        | (224 Bit)       | 82,81 % | (256 Bit) |

Je genauer die lokale Nachbarschaft eines Knotens in der ANT codiert wird, desto mehr Pfade kann GRANT finden. Welche Konfigurationsparameter am besten geeignet sind, hängt somit von dem konkreten Einsatzszenario ab und kann nicht universell beantwortet werden. In Gebieten mit wenig Mobilität ist es ausreichend, die ANT in größeren Zeitintervallen zu aktualisieren, sodass die 32 Byte der ANT-32-8-32 das Netzwerk nur moderat belasten. In hoch mobilen Szenarien ändert sich wiederum die Nachbarschaft schneller, und dementsprechend müssen die ANTs häufiger ausgetauscht werden. In einem solchen Fall kann es sinnvoller sein, eine kleine ANT mit beispielsweise  $a = 4$ ,  $b = 4$  und  $c = 8$  zu verwenden.

Da die Genauigkeit des gespeicherten Winkels sowohl von der Anzahl der Gebiete als auch von der Anzahl der Winkelschritte pro Gebiet abhängt, können nicht alle Ergebnisse direkt miteinander verglichen werden. Eine Genauigkeit von insgesamt 16 Winkelschritten kann beispielsweise durch  $a = b = 4$  oder durch  $a = 16$  und  $b = 0$  erreicht werden. Unabhängig von der Anzahl der Distanzschrte erreicht die zuletzt genannte Variante aufgrund der höheren Anzahl von Einträgen jeweils ein um circa 4,5 Prozentpunkte besseres Ergebnis. Ob diese Verbesserung die Vervielfachung des Speicherbedarfs rechtfertigt, hängt vom konkreten Anwendungsfall ab. Bei gleich bleibender Winkelgenauigkeit verbessert die Verdopplung der Distanzschrte von 8 auf 16 die Ergebnisse um circa 0,8 Prozentpunkte. Die weitere Erhöhung auf 32 Distanzschrte bewirkt hingegen nur noch eine Verbesserung von etwa 0,3 Prozentpunkten.

In den folgenden Simulationen verwendet GRANT eine 6 Byte große ANT mit den Parametern  $a = 8$ ,  $b = 4$  und  $c = 16$ . Diese Konfiguration ist ein guter Kompromiss zwischen dem benötigten Speicherbedarf und den erzielten Ergebnissen. Ein möglicher Vorteil dieser Kombination liegt darin, dass die verwendete ANT den gleichen Speicherbedarf hat wie eine MAC-Adresse und somit in manchen Einsatzszenarien anstatt einer solchen in existierenden Protokollen hinterlegt werden kann. Der IEEE-802.11-Protokoll-Header [80] enthält beispielsweise drei Adressfelder, die abhängig vom Typ des Frames unterschiedlich interpretiert werden. Dabei gibt es mehrere Sonderfälle, in denen eines der drei Felder nicht zwingend notwendig ist und für die Speicherung einer ANT genutzt werden könnte.

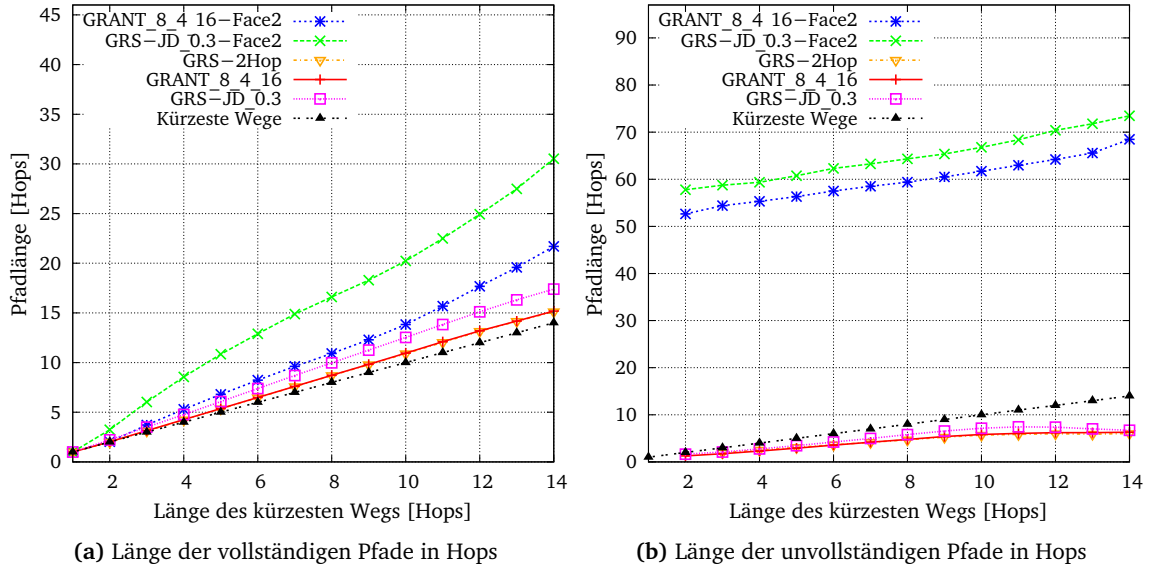
Abbildung 3.25 zeigt die Ergebnisse von GRANT-8-4-16 mit und ohne Überbrückungsverfahren im Vergleich zu GRS-2Hop. Wie im vorherigen Abschnitt beschrieben, benötigt GRS-2Hop eine 2-Hop-Nachbarschaftstabelle, die durch den Austausch von 1-Hop-Nachbarschaftstabellen konstruiert wird. Ausgehend von der durchschnittlichen Anzahl der Knoten innerhalb der Funkreichweite  $N$  verursacht eine vollständige Aktualisierung der Nachbarschaftstabelle ein Datenaufkommen von mindestens  $N \cdot N \cdot 12$  Byte beziehungsweise von  $N \cdot N \cdot 8$  Byte, wenn auf die Übertragung der Identifikatoren verzichtet wird. Aufgrund der Knotenselektion sowie der relativen und verlustbehafteten Codierung der Positionen der Repräsentanten kann das Datenaufkommen bei der Verwendung einer ANT-8-4-16 auf  $N \cdot 6$  Byte gesenkt werden. Trotz der hohen Platzersparnis kann GRANT nahezu zu den Ergebnissen von GRS-2Hop aufschließen. Beispielsweise kann GRS-2Hop für circa 82% der möglichen Kommunikationspartner, die sich auf dem kürzesten Weg etwa 1500 m voneinander entfernt befinden, einen Pfad finden. Durch den Einsatz von GRANT verschlechtert sich das Ergebnis nur um etwa drei Prozentpunkte,



**Abbildung 3.25:** Ergebnisse der Pfadsuche von GRANT im Vergleich zu GRS-2Hop und weiteren Verfahren bei einer Funkreichweite von 300 m und einer Knotendichte von 25,37 Knoten/km.

was in der Relation zu der massiven Reduktion des Datenvolumens ein sehr geringer Wert ist. Bei der betrachteten Kommunikationsdistanz von 1500 m erzielt GRANT im Vergleich zu GRS mit Kreuzungserkennung eine Verbesserung von etwa 30 Prozentpunkten und im Vergleich zu GRS ohne Kreuzungserkennung eine Verbesserung von 41 Prozentpunkten. Im Unterschied zu GRS mit Kreuzungserkennung kann GRANT nicht nur die Ergebnisse von GRS übertreffen, sondern in Kombination mit Face-2 als Überbrückungsstrategie auch deutlich mehr Pfade finden als GRS-Face2. Wenn sich beispielsweise zwei Knoten entlang des kürzesten Wegs etwa 1500 m voneinander entfernt befinden, kann GRANT-Face2 mit einer 98-prozentigen Wahrscheinlichkeit einen Pfad finden und übertrifft somit das Ergebnis von GRS-JD-Face2 um circa fünf Prozentpunkte.

Ein wesentlicher Nachteil der Überbrückung mit Hilfe des Face-2-Verfahrens liegt darin, dass die gefundenen Pfade vergleichsweise lang sind. Des Weiteren produzieren auch die nicht erfolgreichen Überbrückungen sehr lange unvollständige Pfade. Abbildung 3.26 zeigt sowohl die Länge der vollständigen (links) als auch der unvollständigen Pfade (rechts). Die von GRANT und GRS-2Hop gefundenen Pfade sind im Durchschnitt gleich lang und kürzer als die von GRS mit Kreuzungserkennung gefundenen. Die sehr guten Ergebnisse von GRANT führen dazu, dass die Überbrückungsstrategie signifikant seltener eingesetzt werden muss, sodass die durchschnittliche Pfadlänge von GRANT-Face-2 deutlich unter der von GRS-JD-Face-2 liegt. Die Länge der unvollständigen Pfade – also die durchschnittlich zurückgelegte Anzahl von Hops bis zum Abbruch einer Pfadsuche – hängt primär von den gewählten Abbruchbedingungen ab. In den durchgeführten Simulationen wird eine Pfadsuche abgebrochen, wenn ein einzelner



**Abbildung 3.26:** Länge der vollständigen und der unvollständigen Pfade von GRANT und GRS-2Hop bei einer Funkreichweite von 300 m und einer Knotendichte von 25,37 Knoten/km.

Überbrückungsvorgang mehr als 50 Hops benötigt oder der gesamte Pfad länger als 300 Hops wird. In Anbetracht der dargestellten Ergebnisse ist bei allen Kombinationen mit Face-2 die Längenlimitierung der Überbrückungsvorgänge die häufigste Abbruchursache. Die Differenz zwischen der Pfadlänge und den 50 Hops der Abbruchbedingung setzt sich aus der Basisstrategie und den erfolgreichen Überbrückungsvorgängen zusammen. Da GRANT im Vergleich zu GRS-JD kürzere Pfade findet, sinkt auch die durchschnittliche Länge der unvollständigen Pfade in den Varianten mit Überbrückungsstrategie.

#### 3.4.5 Abschätzen der hindernisfreien Umgebung

Der Kreis mit dem betrachteten Knoten  $v$  als Mittelpunkt und der Funkreichweite als Radius wird von der ANT in  $a$  Kreissektoren aufgeteilt. Pro Gebiet wird derjenige Knoten als Repräsentant ausgewählt, welcher den größten Abstand von  $v$  aufweist. Der Empfänger einer ANT weiß somit, dass sich in den codierten Gebieten auf der direkten Strecke vom Sender zum jeweiligen Repräsentanten kein Funkhindernis befindet. Um die hindernisfreie Zone auf einfache Weise abzuschätzen, kann für das Gebiet  $n \in 0, \dots, a-1$  einer ANT der jeweilige Kreissektor auf einen Radius der Länge  $d(v, r_n)$  reduziert werden. Durch die Vereinigung dieser geometrischen Konstrukte kann die hindernisfreie Umgebung um einen Knoten abgeschätzt werden. Dabei werden die Gebiete ohne Repräsentanten ignoriert.

Die anhand der lokalen Nachbarschaft bestimmten ANT's werden in regelmäßigen Abständen zu den benachbarten Knoten übertragen. Auf diese Weise kann die beschriebene Berechnung der hindernisfreien Zone auf alle direkten Nachbarn angewendet und durch eine geometrische

Vereinigung miteinander kombiniert werden. Da pro Gebiet immer der gesamte Kreissektor als hindernisfrei deklariert wird, ist diese Berechnung allerdings nur für  $a \geq 32$  zu empfehlen. Werden in der ANT weniger Gebiete codiert, kann insbesondere in einer Innenstadtumgebung auf eine modifizierte Berechnung zurückgegriffen werden. Hierbei wird anstatt des Kreissektors nur das Gebiet innerhalb eines gewissen Abstands zur Strecke vom betrachteten Knoten zu den jeweiligen Repräsentanten als hindernisfrei betrachtet. Die Berechnung der hindernisfreien Gebiete wird in Algorithmus 3.3 zusammengefasst.

---

**Algorithmus 3.3 :** Die Berechnung der hindernisfreien Gebiete von Knoten  $c$  anhand der ANTs seiner Nachbarn.

---

```
hindernisfreieGebiete :=  $\emptyset$ ;  
for  $n \in NB_{c,1}$  do  
    aktuelleANT := Nachbarschaftstabelle.holeANT( $n$ );  
    for  $h \in [0, a - 1]$  do  
        if aktuelleANT[ $h$ ].codierteLänge > 0 then  
            winkel := decodiereWinkel(ANT[ $h$ ].codierterWinkel);  
            länge := decodiereLänge(ANT[ $h$ ].codierteLänge);  
             $r_h$  := berechneRepräsentantenPosition( $n$ , winkel, länge);  
            aktHindernisfreieGebiet := berechnePufferUmStrecke( $n$ ,  $r_h$ );  
            hindernisfreieGebiete := hindernisfreieGebiete  $\cup$  aktHindernisfreieGebiet;  
return hindernisfreieGebiete;
```

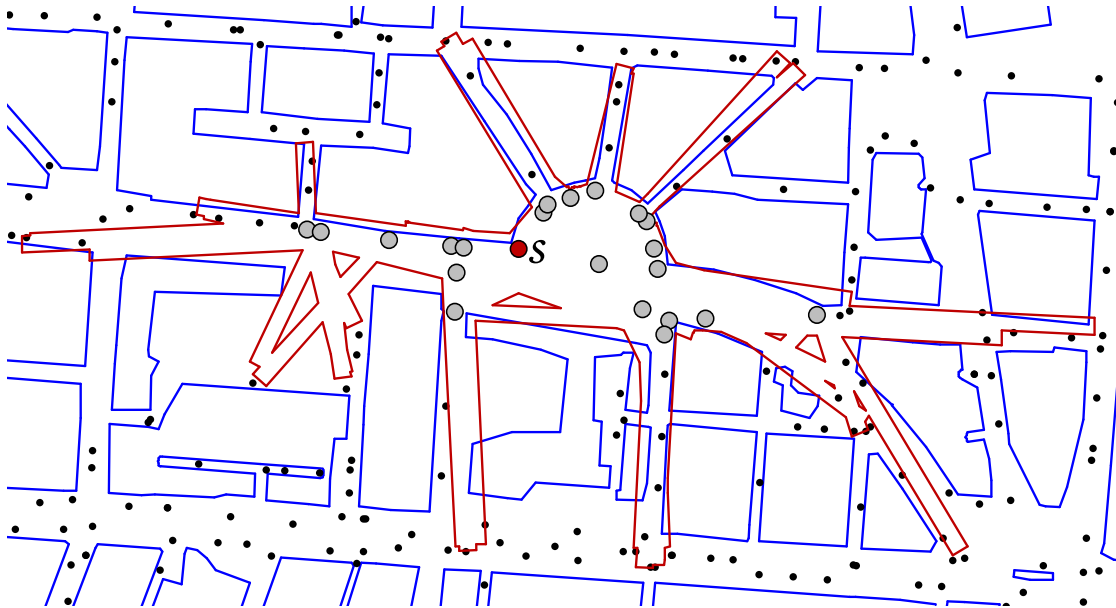
---

Eine exemplarische Abschätzung der hindernisfreien Gebiete anhand der von Knoten  $s$  empfangenen ANTs ist in Abbildung 3.27 zu sehen. Dabei wird das Gebiet in einem Abstand von sieben Metern zu den Strecken von den benachbarten Knoten zu den jeweiligen Repräsentanten als hindernisfrei betrachtet. In einem realen Einsatz ist von einer Mehrwegeausbreitung auszugehen, die dazu führen kann, dass ein Knoten möglicherweise Nachrichten von anderen Knoten empfangen kann, die sich aus seiner Sicht hinter Funkhindernissen befinden. Trotz dieses Umstands kann die beschriebene Abschätzung möglicherweise für einige Anwendungszwecke ausreichend genau sein. Dies gilt insbesondere für Einsatzszenarien, in denen GRANT als Routingverfahren verwendet wird und die benötigten Daten somit ohnehin vorliegen sowie nicht separat ausgetauscht werden müssen.

### 3.5 Vergleich der Überbrückungsverfahren

In dem verwendeten Stadtszenario kann GRS bei einer Knotendichte von 25,37 Knoten pro Kilometer und einer Funkreichweite von 300 m für circa 41 % der möglichen Kommunikationspartner einen Pfad finden. Wie in Abschnitt 3.3.3 dargestellt, können die Ergebnisse von GRS durch den Einsatz einer Überbrückungsstrategie signifikant gesteigert werden. Allerdings ist der durch die Überbrückung entstehende Kommunikationsaufwand beträchtlich, sodass es



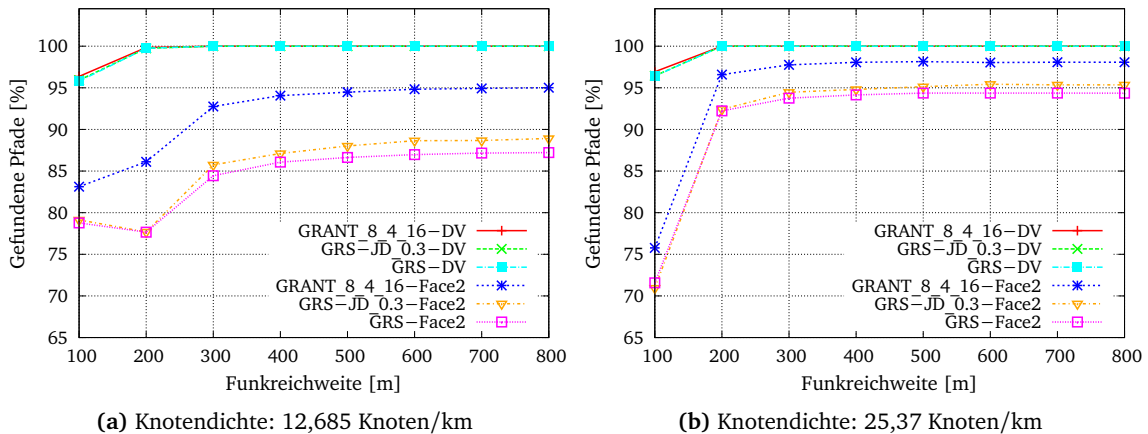


**Abbildung 3.27:** Die Abschätzung der hindernisfreien Gebiete anhand der von Knoten  $s$  empfangenen ANTs (Parameter der verwendeten ANTs:  $a = 8$ ,  $b = 4$  und  $c = 16$ ).

sinnvoll ist, das Basisverfahren zu verbessern. Das hierfür entwickelte Verfahren GRANT übertrifft zwar mit einem Ergebnis von 81 % alle anderen Basisverfahren sehr deutlich, ist aber dennoch auf eine Überbrückungsstrategie angewiesen. Aus diesem Grund werden in diesem Abschnitt die beiden bisher verwendeten Überbrückungsstrategien im Detail analysiert, um abschließend eine Protokollempfehlung zu geben.

Abbildung 3.28 zeigt die Ergebnisse von Face-2 und DV-Routing in Kombination mit GRANT und den beiden GRS-Varianten. Der Prozentsatz der gefundenen Pfade ist in Abhängigkeit von der verwendeten Funkreichweite eingezeichnet. Auf der linken Seite sind die Ergebnisse des Szenarios mit niedrigerer Knotendichte zu sehen, auf der rechten Seite die Ergebnisse des Standardszenarios. Diese Aufteilung wird in den folgenden Abbildungen dieses Abschnitts fortgeführt.

Bei Funkreichweiten unter 300 m ist die Konnektivität des Netzwerks noch eingeschränkt, sodass es zu Beginn zu einem ähnlichen Kurvenverlauf wie in den Ergebnissen ohne Funkhindernisse kommt. Im Vergleich zu den in Abbildung 3.8 dargestellten Basisverfahren können die Überbrückungsverfahren diesen Konnektivitätseffekt allerdings etwas abschwächen. Ab einer Funkreichweite von 300 m findet GRANT mit Face-2 als Überbrückungsstrategie für circa 93 % beziehungsweise 96 % aller möglichen Kommunikationspartner einen Pfad und übertrifft somit deutlich die Ergebnisse der GRS-Varianten. Ab einer Funkreichweite von 200 m finden alle drei Basisverfahren mit Hilfe der DV-Überbrückung für alle Knotenkombinationen einen Pfad.



**Abbildung 3.28:** Die Pfadsuche der Basisverfahren mit Face-2 und DV-Überbrückung in Abhängigkeit von der Funkreichweite und der Knotendichte.

Da eine fehlerhafte Planarisierung bei Face-2 zu einer Endlosschleife führen kann, muss sowohl in einer Simulationsumgebung als auch im realen Einsatz die Länge einer Überbrückung mit Hilfe eines Hop-Zählers limitiert werden. Diese maximale Überbrückungslänge ist somit ein wesentlicher Konfigurationsparameter von Face-2. In der verwendeten Simulationsumgebung können bei DV-Routing zwar keine Endlosschleifen auftreten, aber die Ergebnisse sind stark von dem verwendeten Suchradius des Flutmechanismus abhängig. In allen bisher präsentierten Ergebnissen wurde eine Überbrückung auf 50 Hops limitiert, und es wurde ein Suchradius von 15 Hops verwendet. Beide Werte sind so gewählt, dass im Standardszenario (Funkreichweite 300 m und Knotendichte 25,37 Knoten pro Kilometer) möglichst viele Pfade gefunden werden und gleichzeitig die Simulationsgeschwindigkeit nicht zu stark beeinträchtigt wird. Beide Limitierungen sind auf der Basis von Hops definiert, sodass die verwendete Funkreichweite die räumliche Ausdehnung der Limitierung beeinflusst. Aus diesem Grund ist beispielsweise ein Suchradius von 15 Hops bei einer Funkreichweite von 100 m nicht ausreichend, um alle Pfade zu finden. Aufgrund der Planarisierung ist der Einfluss der Funkreichweite auf Face-2 zwar wesentlich geringer als bei den anderen Verfahren, aber dennoch vorhanden. Die Anzahl der Kommunikationspartner, für die kein Pfad gefunden wird, kann allerdings ab einer Funkreichweite von circa 300 m durch eine Erhöhung des Überbrückungslimits nur noch unwesentlich reduziert werden.

Welches der beiden Überbrückungsverfahren besser geeignet ist, kann anhand der bisher verwendeten Bewertungskriterien nur unzureichend beurteilt werden. So findet etwa DV-Routing unter den verwendeten Simulationsbedingungen – einen ausreichend großen Suchradius vorausgesetzt – immer einen Knoten, der näher am Ziel liegt als derjenige, der die Überbrückung begonnen hat. Ähnlich sieht die Situation bei der durchschnittlichen Pfadlänge aus. Insofern keine Übertragungsfehler oder Ähnliches aufgetreten sind, entsprechen die durch DV-Routing gefundenen Überbrückungspfade immer dem kürzesten Pfad in Bezug auf die Anzahl der Hops. Allerdings steigt bei einer flutbasierten Pfadsuche wie auch beim Aufbau einer Multi-Hop-

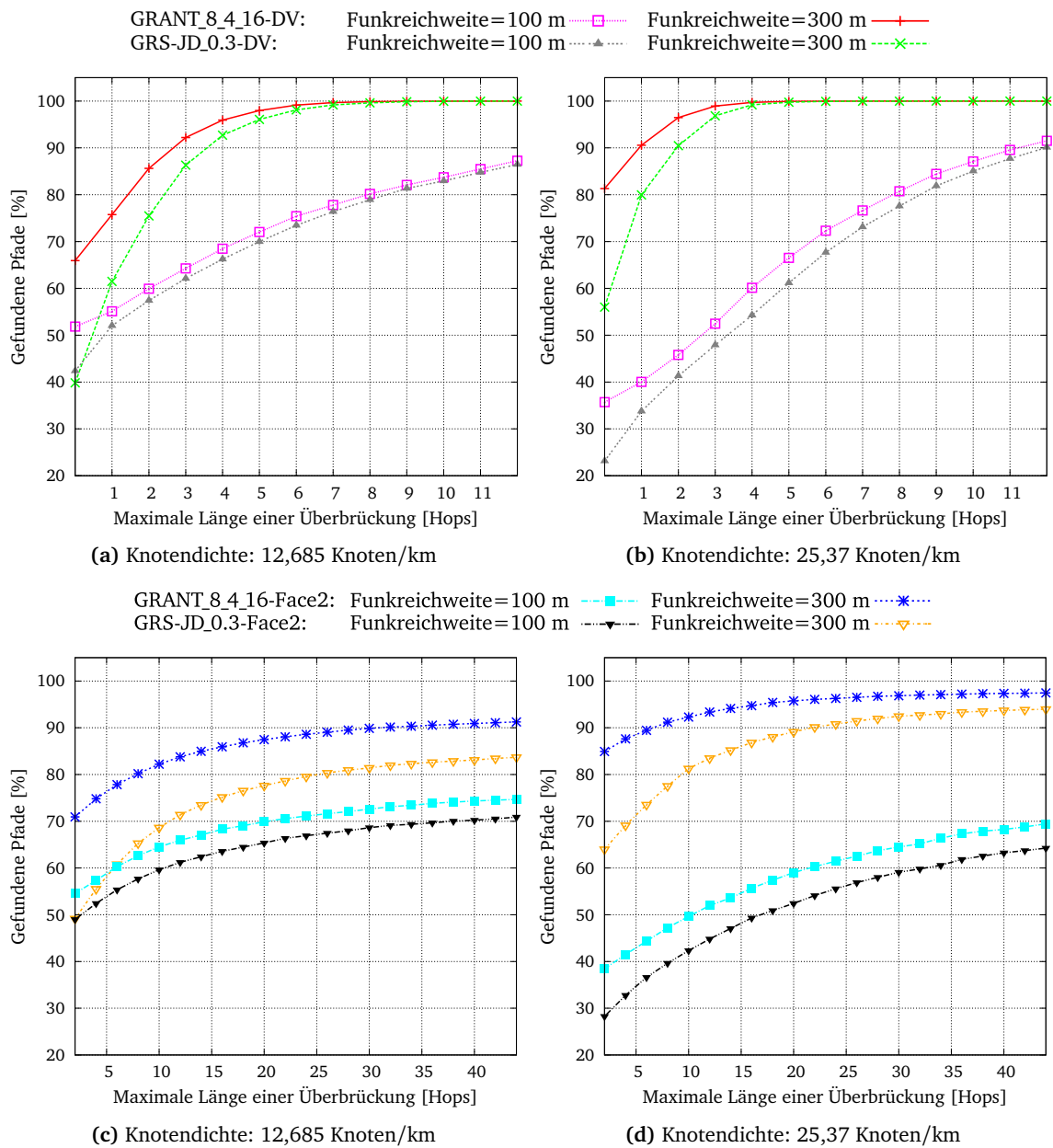
Nachbarschaftstabelle der Kommunikationsaufwand mit jeder Vergrößerung des Radius stark an. Somit sind die Ergebnisse dieser beiden Bewertungskriterien nur unter Berücksichtigung der verwendeten Konfigurationsparameter wie beispielsweise des Suchradius sinnvoll zu beurteilen und vergleichen.

Die Implementierung der DV-Überbrückung sucht in einem Umkreis von 15 Hops nach einem Knoten, der gemäß der verwendeten Metrik näher am Ziel liegt als der Initiator der Suche. Wie bei der Face-2-Implementierung werden auch hier Knoten mit einem näher liegenden Nachbarn berücksichtigt. Falls mehrere Knoten innerhalb des Suchradius für eine Überbrückung geeignet sind, liegt es nahe, denjenigen auszuwählen, der den kleinsten Abstand zu Ziel hat. Um allerdings die Anzahl der rechenintensiven Simulationsdurchläufe auf ein beherrschbares Maß zu beschränken, wählt die DV-Implementierung aus der Menge der geeigneten Kandidaten denjenigen Knoten aus, der mit dem kleinsten Suchradius gefunden werden kann. Durch diese Vorgehensweise kann nach einer erfolgreichen Pfadsuche anhand der längsten aufgetretenen Überbrückung festgestellt werden, ob der Pfad auch bei einem bestimmten Überbrückungslimit gefunden worden wäre. Die von Face-2 gefunden Pfade können auf diese Weise ebenfalls im Nachhinein analysiert werden.

Abbildung 3.29 zeigt den Prozentsatz der gefundenen Pfade beider Überbrückungsverfahren in Abhängigkeit von der maximal erlaubten Überbrückungslänge. Bei beiden Verfahren sind die Ergebnisse bei niedrigerer und höherer Knotendichte jeweils für die Funkreichweiten von 100 m und 300 m eingezeichnet. Wie schon aus Abbildung 3.28 ersichtlich, können die untersuchten Verfahren aufgrund der niedrigen Konnektivität bei einer Funkreichweite von 100 m in beiden Fällen nur vergleichsweise schlechte Resultate erzielen. Dementsprechend kann diese Umgebung als Extremszenario angesehen werden. Im Unterschied hierzu ist die Konnektivität bei beiden Knotendichten ab 300 m ausreichend, um die meisten Pfade zu finden, sodass diese Konfiguration als Standardszenario verwendet wird.

GRANT mit DV-Überbrückung kann im Standardszenario in Abhängigkeit von der Knotendichte bei einem Suchradius von elf beziehungsweise acht Hops für alle Kommunikationspartner einen Pfad finden. Doch schon bei einem Suchradius ab circa sieben beziehungsweise vier Hops ist die Anzahl der nicht erfolgreichen Pfadsuchen zu vernachlässigen. Bei einer Funkreichweite von 100 m reicht der Untersuchungsbereich nicht aus, um alle Pfade zu finden. Allerdings kann auch kaum von einer Überbrückung gesprochen werden, wenn der vollständige Pfad ausschließlich durch den Flutvorgang gefunden wird. Die Kombinationen mit Face-2 als Überbrückungsverfahren können aufgrund der fehlerhaften Planarisierung nicht immer einen Pfad finden, sodass die Ergebnisse unabhängig von der maximalen Überbrückungslänge immer unter 100 % liegen. Bei einer Funkreichweite von 300 m stabilisieren sich die Ergebnisse von GRANT-Face2 bei einem Limit von etwa 40 Hops.

Für eine sinnvolle Bestimmung der geeigneten Konfigurationsparameter und einen abschließenden Vergleich der beiden Verfahren ist aber der Prozentsatz der gefunden Pfade ohne Kenntnis der jeweiligen Kommunikationskosten nicht ausreichend. Diese Kosten hängen sowohl von der Überbrückungshäufigkeit als auch von der konkreten Überbrückungslänge ab. Aus diesem



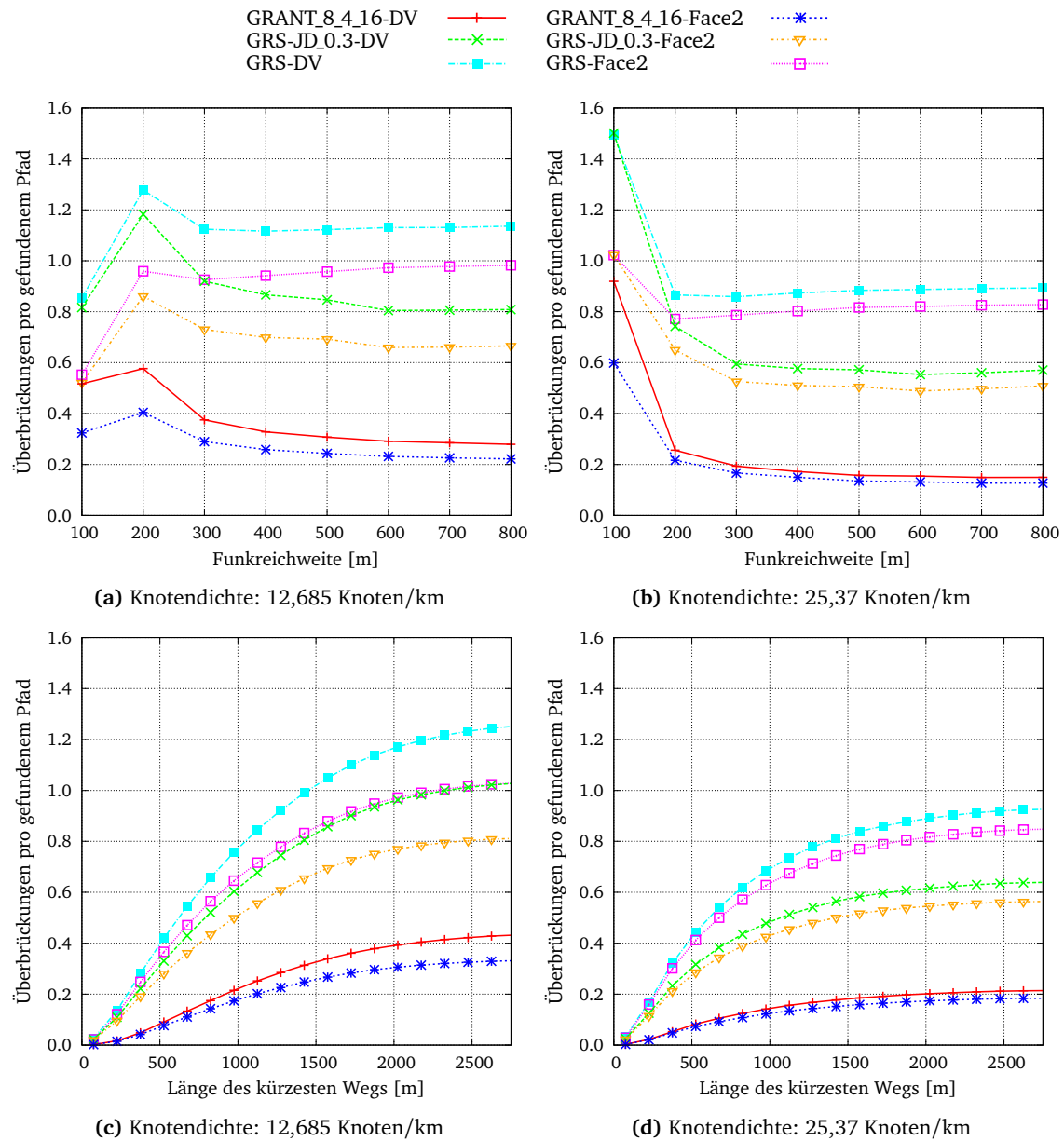
**Abbildung 3.29:** Prozentsatz der gefundenen Pfade bei Verwendung von Face-2 und DV-Routing als Überbrückungsstrategie in Abhängigkeit von der maximal erlaubten Überbrückungslänge.

Grund ist in Abbildung 3.30 die durchschnittliche Anzahl der Überbrückungen pro gefundenem Pfad in Abhängigkeit sowohl von der Funkreichweite als auch von der Kommunikationsdistanz zu sehen. Bei gleich bleibenden Basisverfahren benötigt DV-Routing immer mehr Überbrückungen als Face-2, wobei dieser Umstand sicherlich auch durch die Implementierungsentscheidung beeinflusst wird, immer den kleinstmöglichen Suchradius zu verwenden. Der jeweilige Unterschied zwischen den beiden Überbrückungsverfahren ist in den Szenarien mit niedrigerer Knotendichte stärker ausgeprägt als in denen mit höherer Knotendichte. Bei allen Verfahren stabilisiert sich die Anzahl der benötigten Überbrückungen mit steigender Funkreichweite auf ein nahezu konstantes Niveau. Die Varianten mit GRANT als Basisverfahren greifen mit etwa 0,4 beziehungsweise 0,2 Überbrückungen pro gefundenem Pfad im Durchschnitt deutlich seltener auf die Überbrückungsverfahren zurück als die Kombinationen mit GRS. In der zweiten Zeile der Abbildung sind die Ergebnisse bei einer Funkreichweite von 300 m in Abhängigkeit von der Kommunikationsdistanz gruppiert. Je größer der Abstand zwischen beiden Kommunikationspartnern entlang des kürzesten Pfads ist, desto mehr Überbrückungen sind im Durchschnitt notwendig, um einen Pfad zu finden. Allerdings verläuft der Anstieg der Überbrückungen nicht proportional zur Kommunikationsdistanz und flacht deutlich ab.

Neben der Überbrückungshäufigkeit ist auch deren Länge von Bedeutung; daher wurden in Abbildung 3.31 für die wichtigsten Verfahren die durchschnittlich benötigte Anzahl von Hops pro erfolgreicher Überbrückung eingezeichnet. Wie in der vorherigen Abbildung sind auch hier die Ergebnisse in Abhängigkeit sowohl von der Funkreichweite als auch von der Kommunikationsdistanz gruppiert. Unter der Voraussetzung einer ausreichenden Konnektivität – die ab einer Funkreichweite von etwa 400 m beziehungsweise 300 m erreicht wird – ist die durchschnittliche Überbrückungslänge von allen untersuchten Verfahren nahezu konstant. Auch der Einfluss der Kommunikationsdistanz entlang des kürzesten Pfads ist bei allen Verfahren vergleichsweise gering. Somit ergibt sich aus den letzten beiden Abbildungen, dass mit steigender Kommunikationsdistanz zwar die Wahrscheinlichkeit und Häufigkeit einer Überbrückung steigt, aber ab einer bestimmten Kommunikationsdistanz nicht mehr deren Länge.

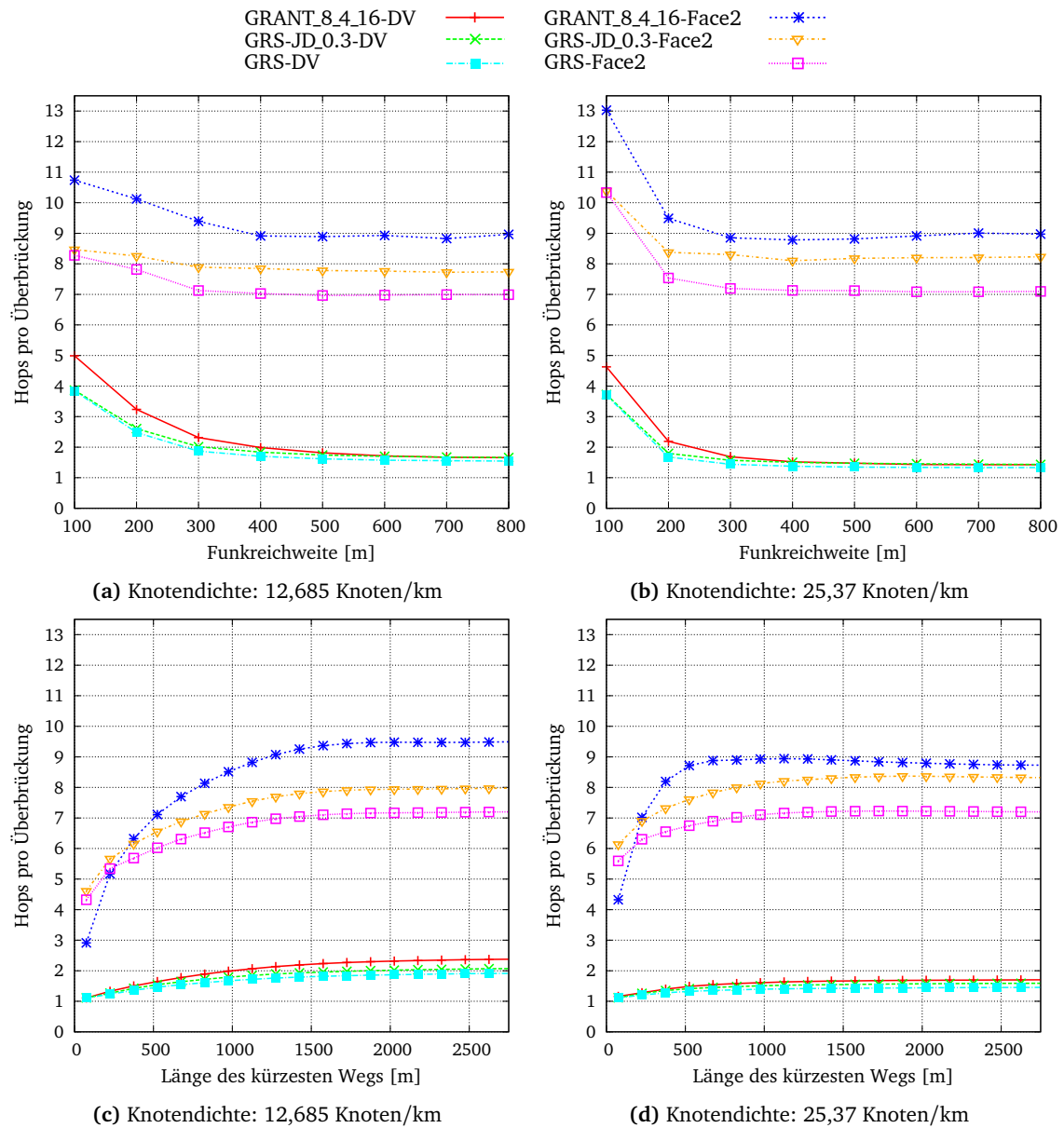
Unabhängig vom jeweiligen Überbrückungsverfahren benötigen die Varianten mit erfolgreicherem Basisverfahren im Durchschnitt mehr Hops pro Überbrückung als die mit weniger erfolgreichem Basisverfahren. Dies begründet sich aus dem Umstand, dass die weniger erfolgreichen Verfahren auch an solchen Stellen auf das Überbrückungsverfahren zurückgreifen müssen, an denen die erfolgreicheren Verfahren gar nicht erst in ein lokales Optimum geraten. Aufgrund dieser zusätzlichen und tendenziell kürzeren Überbrückungen sinkt die durchschnittliche Überbrückungslänge der weniger erfolgreichen Verfahren.

Da Face-2 auf einem planarisierten Graphen mit vergleichsweise kurzen Kanten aufbaut, benötigt dieses Verfahren deutlich mehr Hops pro Überbrückung als DV-Routing. In den Szenarien mit höherer Knotendichte und einer Funkreichweite von 300 m benötigt GRANT mit DV-Routing als Überbrückungsstrategie beispielsweise im Durchschnitt 1,8 Hops pro erfolgreicher Überbrückung. Unter den gleichen Voraussetzungen benötigt die Variante mit Face-2 etwa 9 Hops pro Überbrückung. Durch die Kombination mit den Ergebnissen der vorherigen



**Abbildung 3.30:** Die durchschnittliche Anzahl der Überbrückungen pro gefundenem Pfad in Abhängigkeit von der Funkreichweite und der Kommunikationsdistanz. In der zweiten Zeile der Abbildung sind die Ergebnisse bei einer Funkreichweite von 300 m zu sehen.

### 3.5 Vergleich der Überbrückungsverfahren



**Abbildung 3.31:** Durchschnittliche Anzahl der Hops pro erfolgreicher Überbrückung in Abhängigkeit von der Funkreichweite und der Kommunikationsdistanz. In der zweiten Zeile der Abbildung sind die Ergebnisse bei einer Funkreichweite von 300 m zu sehen.



Abbildung kann die durchschnittliche Anzahl der Hops im Überbrückungsmodus pro gefundenem Pfad abgeschätzt werden. GRANT-DV benötigt bei 1,8 Hops pro Überbrückung etwa 0,2 Überbrückungen pro gefundenem Pfad, was in etwa 0,36 Hops im Überbrückungsmodus pro gefundenem Pfad entspricht. Demgegenüber ergibt sich aus den 9 Hops pro Überbrückung und den etwa 0,18 Überbrückungen pro gefundenem Pfad, dass die Variante mit Face-2 etwa 1,62 Hops pro gefundenem Pfad benötigt.

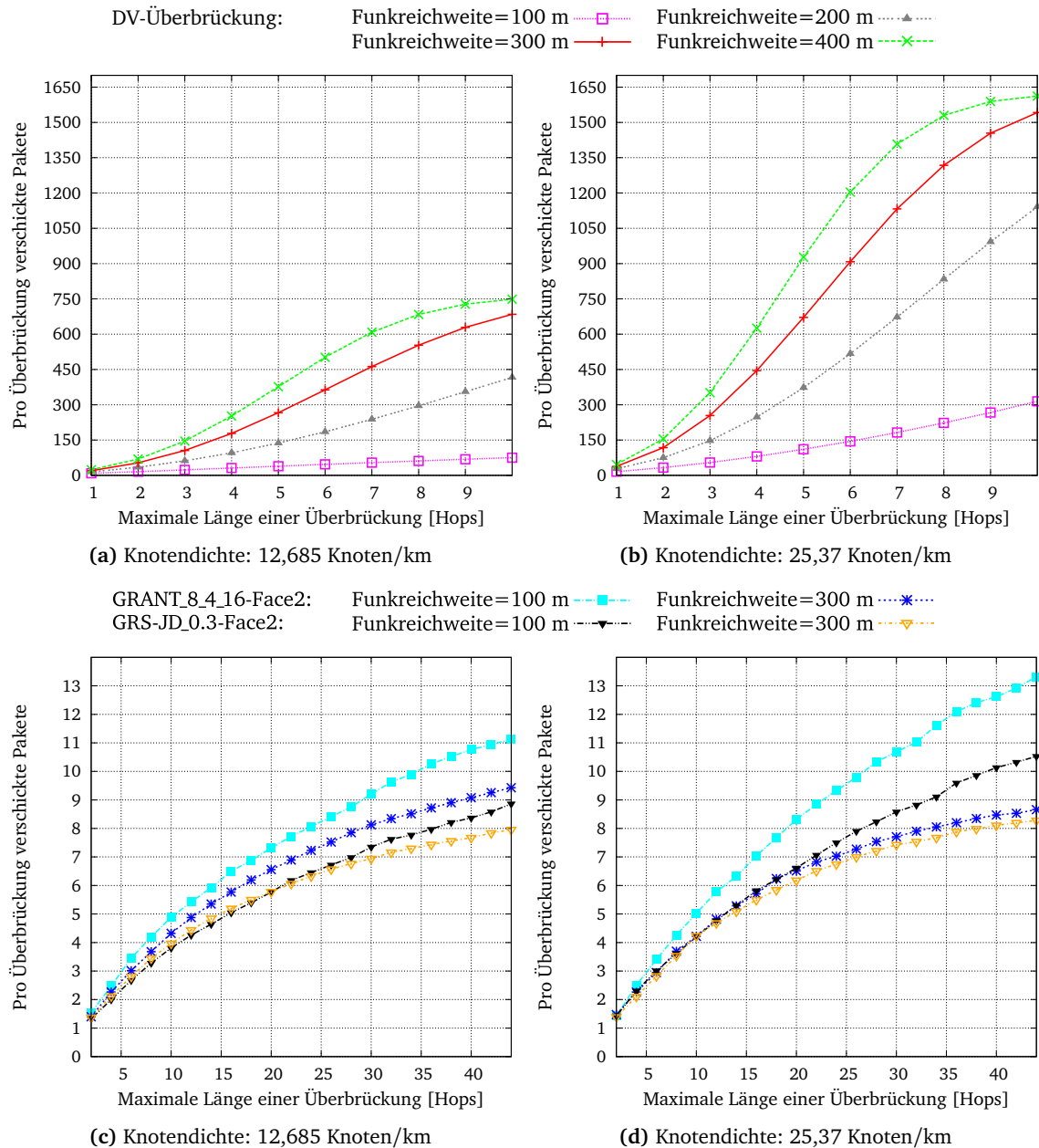
Die Ergebnisse der beiden letzten Abbildungen beziehen sich auf einen Suchradius von 15 Hops und eine maximale Überbrückungslänge von 50 Hops. Der Kurvenverlauf und die besprochenen Abhängigkeiten von Funkreichweite und Kommunikationsdistanz sind von diesen Konfigurationsparametern weitgehend unabhängig. Allerdings wird die durchschnittliche Anzahl der Überbrückungen und deren Länge, also die Position der Kurve auf der y-Achse, direkt von diesen Parametern beeinflusst. Durch eine Reduktion der maximal erlaubten Überbrückungslänge sinkt beispielsweise bei den Varianten mit Face-2 sowohl die im Durchschnitt benötigte Anzahl der Hops pro Überbrückung als auch die Anzahl der gefundenen Pfade. Da allerdings die Anzahl der Überbrückungen und deren Länge noch nicht die Kosten für die flutbasierte Suche von DV-Routing enthalten, wird die Abhängigkeit dieser beiden Kenngrößen von der maximalen Überbrückungslänge hier nicht dargestellt. Stattdessen wurde in Abbildung 3.32 eingezeichnet, wie viele Pakete – abhängig von der maximal erlaubten Überbrückungslänge – im Durchschnitt pro erfolgreicher Überbrückung verschickt werden.

Bei einer flutbasierten Suche wird ein Paket mit der Suchanfrage zu jedem Knoten innerhalb des Suchradius übertragen. Typischerweise wird eine derartige Suche gestartet, indem der Initiator ein entsprechendes Paket per Broadcast an seine direkten Nachbarn schickt. Jeder Empfänger überprüft, ob er selbst oder einer seiner Nachbarn besser geeignet ist als der Initiator. Falls er keinen geeigneten Knoten kennt, das Suchlimit noch nicht erreicht wurde und er zuvor noch keine Kopie der Suchanfrage empfangen hat, leitet er das Paket wiederum an alle seine Nachbarn weiter. Falls er aber einen geeigneten Knoten kennt, schickt der Empfänger eine entsprechende Antwort zurück zum Initiator und verwendet dafür die auf dem Hinweg der Suchanfrage gespeicherten Routingtabelleneinträge. Durch das Zählen der Nachbarn innerhalb des Suchradius kann die Anzahl der während der Suche übertragenen Pakete in etwa abgeschätzt werden. Zu dieser Summe werden noch die Pakete entlang des jeweils entdeckten Pfads addiert. Da die Anzahl der Pakete für die Überbrückungssuche unabhängig vom verwendeten Basisverfahren ist und die durchschnittlichen Überbrückungslängen aller Basisverfahren in etwa gleich sind, werden in Abbildung 3.32 die kombinierten Ergebnisse aller DV-Varianten gezeigt.

Aus Gründen der Übersichtlichkeit werden bei Face-2 nur die Ergebnisse von GRANT und GRS mit Kreuzungserkennung bei einer Funkreichweite von jeweils 100 m und 300 m präsentiert. Im Unterschied zu den Ergebnissen von Face-2 ist die Anzahl der Pakete pro flutbasierter Suche stark von der Funkreichweite abhängig, sodass die kombinierten DV-Routing-Ergebnisse auch für die Funkreichweiten von 200 m und 400 m dargestellt sind. In der freien Ebene ist die Paketanzahl der flutbasierten Suche exponentiell von der Funkreichweite anhängig. Durch die



### 3.5 Vergleich der Überbrückungsverfahren



**Abbildung 3.32:** Durchschnittliche Anzahl der verschickten Pakete pro erfolgreicher Überbrückung in Abhängigkeit von der maximal erlaubten Überbrückungslänge.

Funkhindernisse und die räumliche Begrenzung des Simulationsgebiets wird die Paketanzahl im simulierten Szenario mit steigendem Suchradius stark begrenzt, wobei die letztgenannte Ursache in einem realen Einsatzgebiet möglicherweise nicht gegeben wäre. Für das Szenario mit höherer Knotendichte ist bei einer Funkreichweite von 300 m allerdings bereits ein Suchradius von 5 Hops ausreichend, um alle Pfade zu finden, sodass die räumliche Begrenzung des Simulationsgebiets die relevanten Ergebnisse noch nicht signifikant beeinflusst.

Bei einem Suchradius von 3 Hops und einer Funkreichweite von 300 m werden abhängig von der Knotendichte circa 100 beziehungsweise 375 Pakete pro Überbrückung verschickt und gleichzeitig 91 % beziehungsweise 99 % aller Pfade gefunden. Im Vergleich hierzu verschickt Face-2 im Einsatz mit GRANT bei einer maximal erlaubten Überbrückungslänge von 15 Hops und gleich bleibender Funkreichweite in beiden Fällen durchschnittlich etwa 5,5 Pakete pro Überbrückung und findet hierbei für 85 % beziehungsweise 95 % aller möglichen Kommunikationspartner einen Pfad. Auch wenn bei GRANT der Unterschied nicht mehr allzu groß ist, so benötigt DV-Routing im Durchschnitt doch mehr Überbrückungen pro gefundenem Pfad als Face-2. Dementsprechend verschickt Ersteres nicht nur mehr Pakete pro Überbrückung, sondern auch pro gefundenem Pfad. Mit der hohen Paketanzahl steigt zwar das Risiko für Paketkollisionen, allerdings ergibt sich hieraus nicht unmittelbar die insgesamt benötigte Bandbreite. Während bei Face-2 die eigentlichen Nutzdaten direkt bei der Überbrückungssuche mitgeführt werden, leitet DV-Routing die Nutzdaten erst weiter, nachdem die flutbasierte Suche einen Überbrückungspfad gefunden hat. Dementsprechend sind die gefluteten Pfadanfragen und auch die im Erfolgsfall zurückgeschickten Pfadantworten vergleichsweise klein.

Die Pfadanfragen von AODV werden *Route Request Messages* genannt, die Pfadantworten *Route Reply Messages*. Diese Pakete enthalten unter anderem Statusinformationen, Sequenznummern und die IP-Adressen von Ziel und Absender. Gemäß RFC 3561 [120] sind diese Pakete 24 Byte beziehungsweise 20 Byte groß. Für eine positionsbasierte Erweiterung werden zusätzlich noch die Positionen von Ziel und Absender benötigt. Positionen werden meist in Form des Breiten- und Längengrads hinterlegt, die jeweils 4 Byte benötigen. Dementsprechend vergrößern sich die Pakete jeweils um 16 Byte auf 40 Byte beziehungsweise 36 Byte. Zusätzlich werden pro Paket noch die Daten der tiefer liegenden Netzwerkschichten übertragen. Im Folgenden wird von einem Einsatz mit IEEE 802.11 [80] ausgegangen, sodass sich die Pakete um weitere 28 Byte vergrößern. Dementsprechend benötigt jede Pfadanfrage insgesamt 68 Byte und jede Pfadantwort 64 Byte.

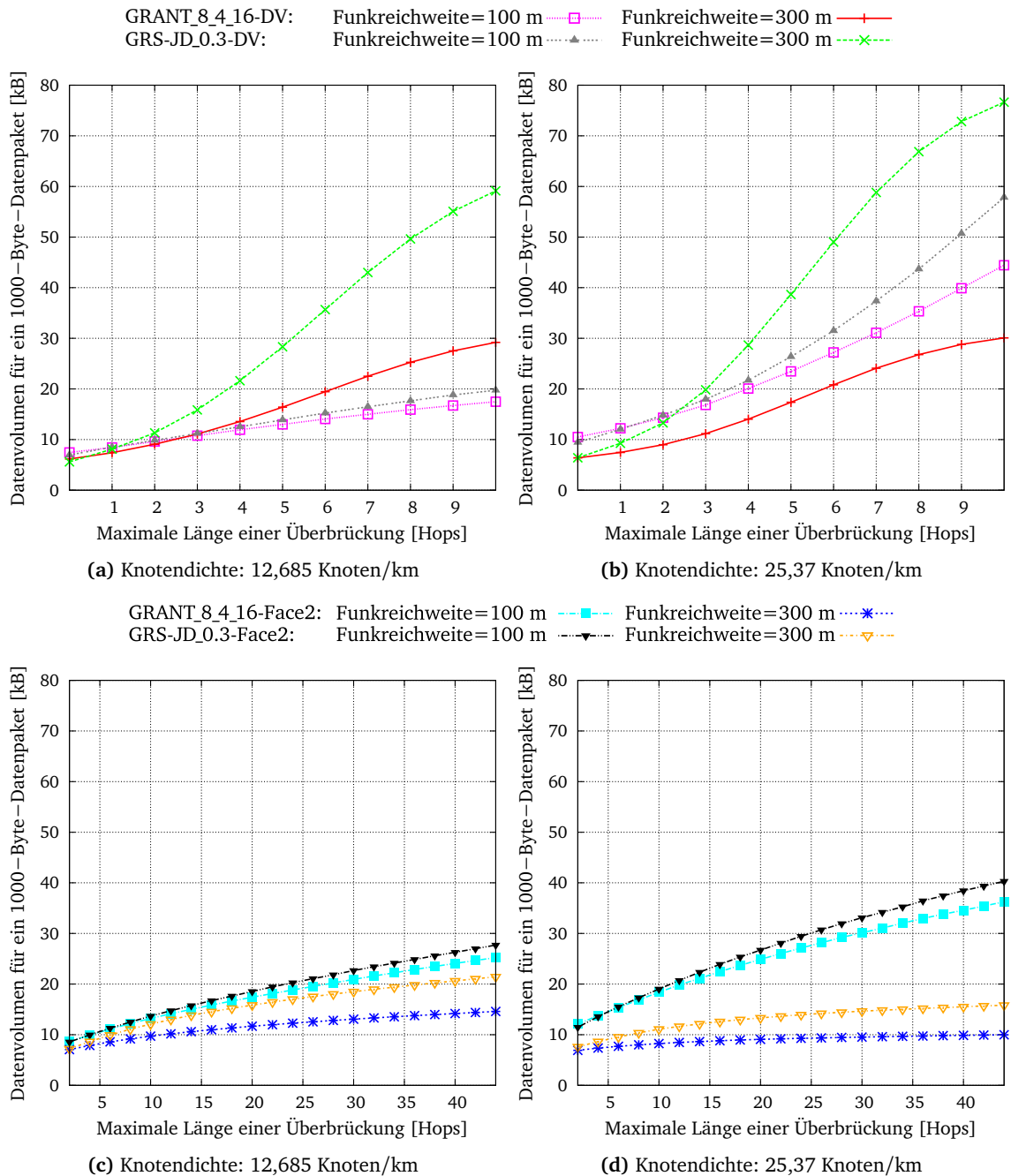
Das während einer erfolgreichen Überbrückung übertragene Datenvolumen setzt sich bei DV-Routing aus den innerhalb des Suchradius verschickten Pfadanfragen, den zurückgeschickten Pfadantworten und den entlang des gefunden Überbrückungspfads gesendeten Datenpaketen zusammen. Bei Face-2 entfällt die flutbasierte Suche, sodass nur die Übertragungskosten des Datenpakets entlang des gefundenen Pfads anfallen. Die Gesamtkosten für die Zustellung eines Datenpakets beinhalten die Kosten des Basisverfahrens und die der jeweils notwendigen Überbrückungen. Kann ein Datenpaket aufgrund einer erfolglosen Pfadsuche nicht zugestellt werden, fallen dennoch Übertragungskosten an. Neben den Kosten des Basisverfahrens und

den Kosten der möglicherweise vorhandenen erfolgreichen Überbrückungen beinhalten diese Gesamtkosten auch die Kosten einer erfolglosen Überbrückung. Im Fall von Face-2 bedeutet dies, dass das Datenpaket während der erfolglosen Überbrückung so oft übertragen wird, wie es die maximal erlaubte Überbrückungslänge gestattet. Im Vergleich hierzu fallen bei DV-Routing zwar die Kosten der flutbasierten Suche an, aber keine Übertragungskosten für das Datenpaket. In Abbildung 3.33 wurden die durchschnittlichen Übertragungskosten für den Versuch, ein Datenpaket mit 1000 Byte Nutzdaten zuzustellen, eingezeichnet. Die präsentierten Ergebnisse beziehen sich immer auf das gesamte Szenario mit allen jeweils möglichen Kommunikationspartnern.

In Bezug auf die Anzahl der gefunden Pfade und die resultierenden Kosten sind ein Suchradius von 3 Hops für GRANT-DV und eine Limitierung der Überbrückungslänge auf 15 Hops für GRANT-Face2 ein guter Kompromiss für ein Szenario mit einer Funkreichweite von 300 m. Bei dieser Konfiguration benötigt GRANT-DV in den Szenarien mit höherer Knotendichte im Durchschnitt etwa 11 Kilobyte, um 1000 Byte Nutzdaten zuzustellen. Im Vergleich hierzu überträgt GRANT mit Face-2 und einer maximal erlaubten Überbrückungslänge von 15 Hops unter gleichen Umständen im Durchschnitt circa 9 Kilobyte. Dies bedeutet, dass sich das reale Datenaufkommen pro Pfad nur um etwa 22 % unterscheidet, obwohl GRANT-DV pro Überbrückung mehr als 68-mal so viele Pakete verschickt wie GRANT-Face2. In beiden Fällen findet GRANT ohne Überbrückungsmechanismus für 81,05 % aller möglichen Kommunikationspartner einen Pfad. Da GRANT-DV bei einem Suchradius von 3 Hops insgesamt circa 99 % aller Pfade findet, ist bei den verbleibenden 17,95 % mindestens eine erfolgreiche Überbrückung notwendig, um das Ziel zu finden. GRANT-Face2 findet bei einer maximal erlaubten Überbrückungslänge von 15 Hops für 95 % aller Knotenkombinationen einen Pfad und greift somit in etwa 13,95 % aller Fälle auf Face-2 zurück. In rund 5 % aller Fälle wird die Pfadsuche nach einer erfolglosen, auf 15 Hops begrenzten Überbrückung abgebrochen.

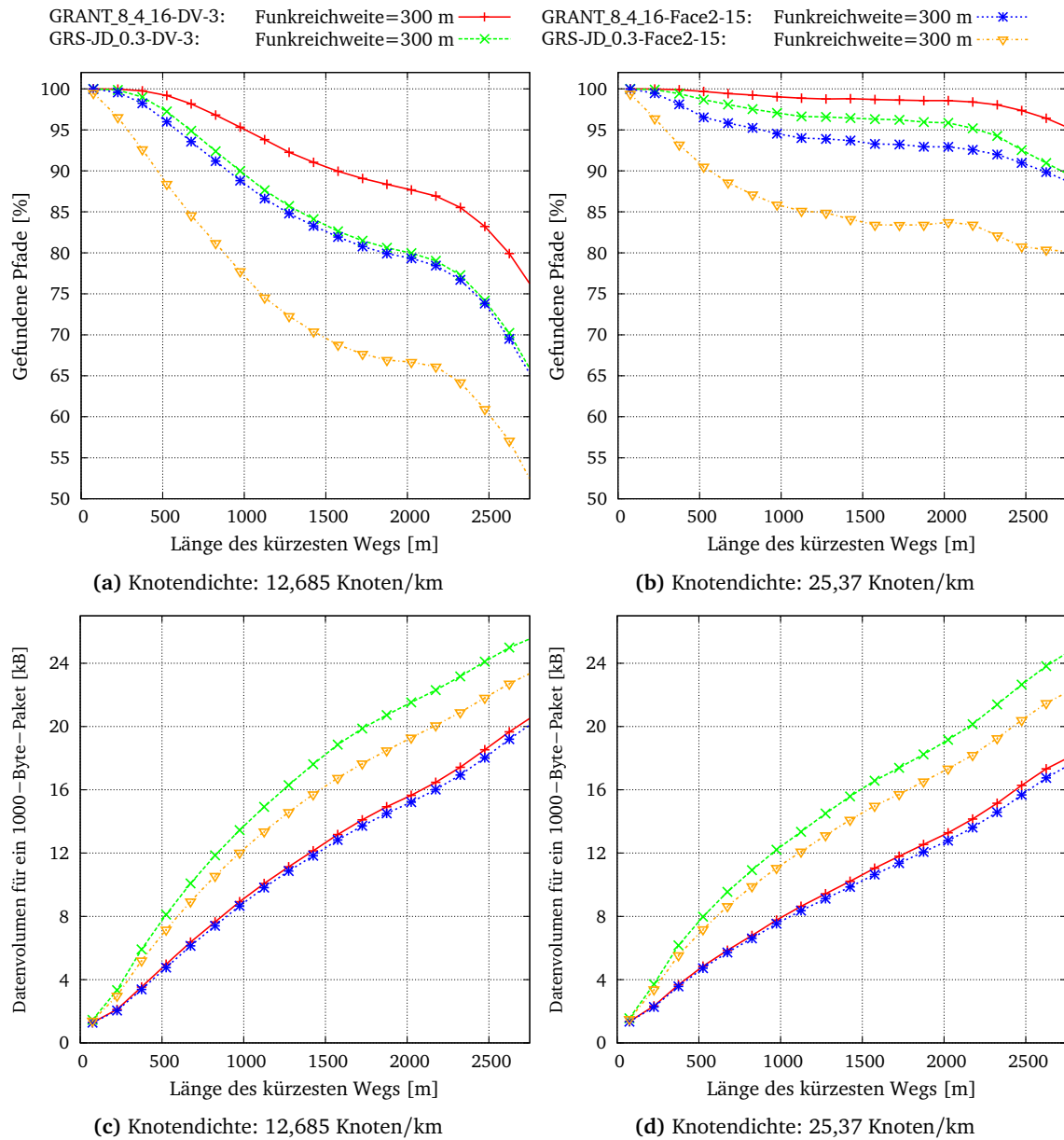
Bei einer Funkreichweite von 100 m ist die Konnektivität nicht ausreichend, um zufriedenstellende Ergebnisse zu erzielen. Die Resultate können aber zumindest durch die Wahl von deutlich höheren Limits etwas verbessert werden. Bei einem großzügigen Suchradius von 10 Hops verursacht GRANT-DV ein Datenaufkommen von etwa 18 Kilobyte (niedrigere Knotendichte) beziehungsweise 44 Kilobyte (höhere Knotendichte) und findet hierbei immerhin für 82 % beziehungsweise 88 % aller Kommunikationspartner einen Pfad. Im direkten Vergleich hierzu überträgt GRANT-Face2 bei einem Überbrückungslimit von 45 Hops Pakete in einem Volumen von etwa 25 Kilobyte beziehungsweise 36 Kilobyte und findet hierbei circa 75 % beziehungsweise 69 % aller Pfade. Bei diesen Überbrückungslimits erzielt GRANT-DV in den Szenarien mit niedrigerer Knotendichte bessere Resultate als GRANT-Face2, verursacht aber deutlich höherer Kosten, wenn die Knotendichte steigt. Wie dieses Beispiel zeigt, sind die Ergebnisse stark von den Rahmenbedingungen abhängig, sodass es nicht möglich ist, eine universell gültige Konfigurationsempfehlung zu geben.

Abbildung 3.34 zeigt die Simulationsergebnisse der beiden Überbrückungsverfahren im Standardszenario mit der empfohlenen Limitierung der Überbrückungslänge auf 3 beziehungs-



**Abbildung 3.33:** Durchschnittlich anfallendes Datenvolumen für den Versuch, ein Datenpaket mit 1000 Byte Nutzdaten zuzustellen, abhängig von der maximal erlauben Überbrückungslänge.

### 3.5 Vergleich der Überbrückungsverfahren



**Abbildung 3.34:** Simulationsergebnisse der untersuchten Überbrückungsverfahren mit der empfohlenen Limitierung der Überbrückungslänge auf 3 bzw. 15 Hops.

weise 15 Hops. Die Ergebnisse sind in Abhängigkeit vom kürzesten Pfad zwischen den beiden jeweiligen Kommunikationspartnern gruppiert. In den Szenarien mit höherer Knotendichte bleibt der Prozentsatz der Kommunikationspartner, für die GRANT mit dem jeweiligen Überbrückungsverfahren einen Pfad findet, über die gesamte Kommunikationsdistanz hinweg weitgehend stabil. Bei niedrigerer Knotendichte verschlechtern sich allerdings die Ergebnisse beider Verfahren mit steigender Kommunikationsdistanz deutlich. Des Weiteren kann in diesen Fällen ein näherungsweise paralleler Verlauf der beiden Kurven des jeweiligen Basisverfahrens beobachtet werden.

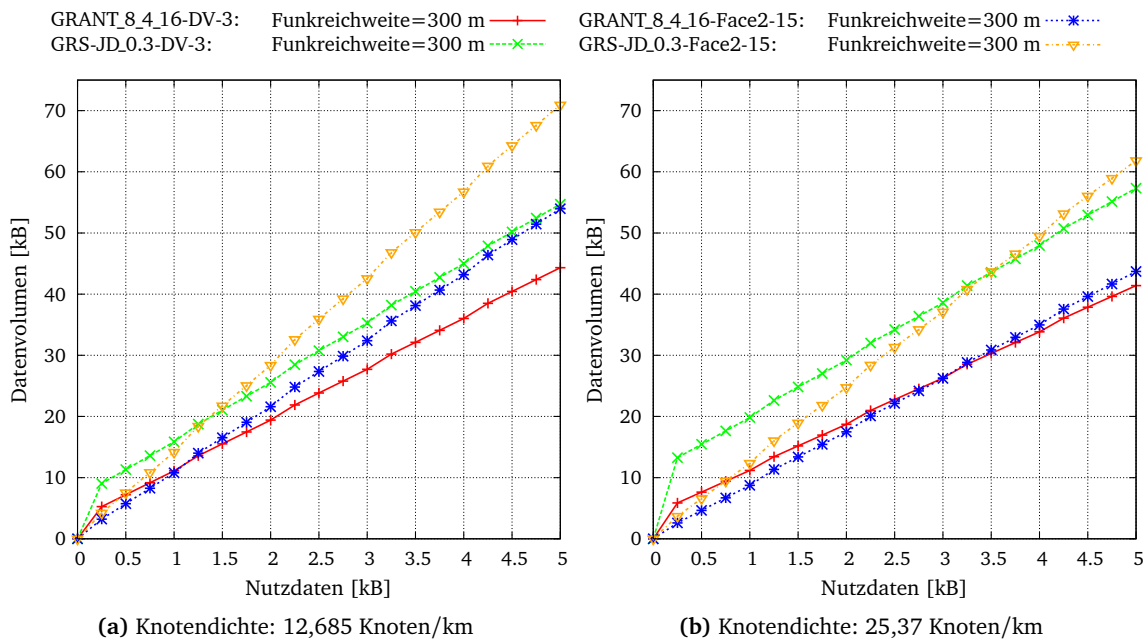
Wie in Abbildung 3.30 und 3.31 dargestellt, steigt die Anzahl der benötigten Überbrückungen mit der Kommunikationsdistanz, während die durchschnittliche Überbrückungslänge nahezu konstant bleibt. Aufgrund dieses Sachverhalts und der Limitierung der Überbrückungslänge steigen die Übertragungskosten von 1000 Byte Nutzdaten nur proportional zur Kommunikationsdistanz. Die in der zweiten Zeile von Abbildung 3.34 gezeigte leichte Abflachung des Datenvolumens im Kurvenverlauf ist auf die Reduktion der gefundenen Pfade zurückzuführen. Wenn beispielsweise zwischen zwei Kommunikationspartnern nur ein komplexer Pfad existiert, der von dem jeweiligen Verfahren nicht gefunden werden kann, dann sind die Kosten umso niedriger, je früher die Pfadsuche abgebrochen wird. Dementsprechend verringert sich der Anstieg des durchschnittlichen Datenaufkommens umso stärker, je mehr Pfade prozentual nicht gefunden werden können.

Bei der verwendeten Limitierung der Überbrückungslänge sind die Kosten beider GRANT-Varianten bei beiden Knotendichten ähnlich hoch, wenngleich sich die Anzahl der jeweils gefundenen Pfade deutlich unterscheidet. Sollen beispielsweise 1000 Byte Nutzdaten in Form eines einzelnen Datenpakets zwischen zwei Knoten mit einer Kommunikationsdistanz von 2000 m ausgetauscht werden, so benötigen beide GRANT-Varianten – abhängig von der Knotendichte – ein Datenvolumen von circa 16 Kilobyte beziehungsweise 13 Kilobyte. Da Face-2 versucht, den Überbrückungspfad durch Weiterleiten des Datenpakets gemäß der Rechte-Hand-Regel zu finden, benötigen Basisverfahren mit Face-2 als Überbrückungsstrategie für die Übertragung eines zweiten Datenpakets – bei gleich bleibender Topologie – exakt das gleiche Datenvolumen wie für das erste Paket. Im Unterschied dazu werden die von der flutbasierten Suche gefundenen Überbrückungspfade verteilt in den Routingtabellen der entsprechenden Knoten hinterlegt und können mehrfach verwendet werden. Bei einer Übertragungsgeschwindigkeit von beispielsweise 54 Megabit pro Sekunde verläuft die Übermittlung von mehreren unmittelbar hintereinander verschickten Paketen derart schnell, dass in den meisten Fällen die in der Zwischenzeit möglicherweise auftretende topologische Veränderung die gefundenen Überbrückungen noch nicht beeinträchtigt.

Abbildung 3.35 zeigt das im Durchschnitt benötigte Übertragungsvolumen in Abhängigkeit von der zu sendenden Nutzdatenmenge. Falls mehr als 1000 Byte Nutzdaten gesendet werden sollen, wird die zu sendende Datenmenge auf mehrere Pakete aufgeteilt, wobei die Kosten der flutbasierten Überbrückungen nur für das erste Paket berücksichtigt werden. In der Abbildung werden die Ergebnisse in 250-Byte-Schritten dargestellt und wurden der Übersichtlichkeit hal-

### 3.5 Vergleich der Überbrückungsverfahren

ber zu einer Kurve verbunden, sodass beispielsweise die Interpolation zwischen 0 und 250 Byte nicht der realen Datenmenge entspricht. Der nicht exakt gleich bleibende lineare Kurvenverlauf kommt dadurch zustande, dass sich an Stellen, an denen die Paketanzahl steigt, das Verhältnis zwischen Nutzdaten und Protokolldaten in den Paketköpfen sprunghaft ändert.



**Abbildung 3.35:** Durchschnittlich anfallendes Datenvolumen für den Versuch, eine bestimmte Nutzdatenmenge zuzustellen. Falls die zu sendende Datenmenge die Begrenzung von 1000 Byte pro Paket übersteigt, wird sie auf mehrere Pakete aufgeteilt.

Aufgrund der Zusammensetzung der Gesamtmenge der Pfade, die beispielsweise auf der linken Seite der Abbildung 3.25 zu sehen sind, benötigen die in Abbildung 3.35 dargestellten Verfahren im Durchschnitt deutlich weniger Datenvolumen als in dem zuvor besprochenen Szenario mit einer Kommunikationsdistanz von 2000 m. In der Simulationsumgebung mit höherer Knotendichte benötigen GRANT-DV und GRANT-Face2 für das erste Paket mit 1000 Byte Nutzdaten etwa 11,2 Kilobyte beziehungsweise 8,8 Kilobyte und für jedes weitere Paket 7,6 Kilobyte beziehungsweise weiterhin 8,8 Kilobyte. Aus diesem Grund ist das Gesamtvolumen der Variante mit Face-2 ab etwa dem dritten Paket höher als das von GRANT-DV. Aufgrund des Kommunikationsbedarfs heutiger Anwendungen ist in vielen Fällen von einem hohen Datenvolumen innerhalb kürzester Zeit auszugehen, sodass in diesen Fällen die unterschiedliche Steigerung der Übertragungskosten das Ergebnis stark zugunsten der Variante mit DV-Routing verschieben würde. Dauert eine Übertragung allerdings so lange, dass währenddessen die gefundenen Überbrückungspfade ungültig werden, so müssen bei DV-Routing auch die entsprechenden Reparaturkosten für die Routingtabellen berücksichtigt werden.



Neben den eigentlichen Überbrückungspfaden werden während der flutbasierten Suche auch andere Pfade gespeichert. Beispielsweise merken sich auch diejenigen Knoten, die an der Überbrückung nicht beteiligt sind, von welchen Nachbarn sie die Anfrage erhalten haben und welcher Knoten die Anfrage initiiert hat. Diese Zusatzinformationen können genutzt werden, um andere Suchanfragen schneller zu beantworten oder gar zu vermeiden. Dieser Umstand macht es wahrscheinlich, dass die durchschnittlichen Kosten einer Überbrückung mit DV-Routing bei einer Erhöhung der gleichzeitig aktiven Kommunikationspartner sinken, wohingegen dieser Aspekt die Kosten von Face-2 nicht beeinflusst.

Die wichtigsten Simulationsergebnisse der beiden Überbrückungsverfahren mit der empfohlenen Limitierung der Überbrückungslänge im Zusammenspiel mit GRANT und GRS-JD wurden in Tabelle 3.3 noch einmal zusammengefasst. Die dargestellten Ergebnisse beziehen sich auf das Szenario mit einer Funkreichweite von 300 m und einer Knotendichte von 25,37 Knoten pro Kilometer. Bei allen Werten handelt es sich um Durchschnittswerte für alle Knotenkombinationen des gesamten Szenarios. Die Länge des kürzesten Pfads zwischen den jeweiligen Kommunikationspartnern ist nicht gleich verteilt, sondern hängt direkt von den topologischen Bedingungen und der räumlichen Begrenzung des Simulationsgebiets ab. Doch auch in einem realen Einsatz müsste aus praktischen Gründen die maximal erlaubte Kommunikationsdistanz beschränkt werden, weil ansonsten die verfügbare Bandbreite und die auftretende Latenz ab einem bestimmten Abstand eine Nutzung nahezu unmöglich machen würden. Da die real auftretenden Kommunikationsstränge des Weiteren auch von den verwendeten Anwendungen und den genutzten Protokollen abhängen, entsprechen die präsentierten Ergebnisse zwar nicht den real zu erwartenden Durchschnittswerten, sind aber dennoch für den direkten Vergleich der beiden Überbrückungsverfahren geeignet.

In die Tabelle wurden die durchschnittlichen Pfadlängen sowohl in Hops als auch in Metern eingetragen. Face-2 und DV-Routing benötigen mit GRANT pro gefundem Pfad im Durchschnitt 0,15 beziehungsweise 0,20 Überbrückungen mit einer durchschnittlichen Länge von 5,5 beziehungsweise 1,6 Hops. Doch trotz der deutlich höheren Anzahl von Hops ist der pro Überbrückung im Durchschnitt zurückgelegte Weg bei Face-2 kleiner als bei DV-Routing. Dieser Umstand ist auf die bei Face-2 notwendige Planarisierung zurückzuführen, welche tendenziell die längsten Kanten des Netzwerkgraphen ignoriert und somit die durchschnittlich zurückgelegte Strecke pro Hop reduziert. Die geringfügigen Unterschiede zwischen den zuvor besprochenen Werten und den in Abbildung 3.30 sowie Abbildung 3.31 präsentierten Ergebnissen sind auf die unterschiedlichen Limitierungen der Überbrückungslänge zurückzuführen.

Abgesehen von den verwendeten Funkhindernissen und der zugrunde liegenden Straßentopologie werden die Simulationsergebnisse im Wesentlichen von der Knotendichte, der Funkreichweite und der Kommunikationsdistanz beeinflusst. In Tabelle 3.4 wird der tendenzielle Einfluss dieser Faktoren bewertet. Im Fall der Knotendichte werden hierfür die Ergebnisse bei einer Funkreichweite von 300 m herangezogen. Mit steigender Knotendichte erhöhen sich die Wahlmöglichkeiten der Routingverfahren, sodass tendenziell für mehr Kommunikationspartner ein Pfad gefunden werden kann. Die Planarisierungsproblematik von Face-2 kann aller-



### 3.5 Vergleich der Überbrückungsverfahren

**Tabelle 3.3:** Übersicht der Simulationsergebnisse von Face-2 und DV-Routing mit der empfohlenen Limitierung der Überbrückungslänge auf 15 bzw. 3 Hops bei einer Funkreichweite von 300 m und einer Knotendichte von 25,37 Knoten/km.

|  | GRANT              |                    | GRS-JD_0.3          |                    |
|--|--------------------|--------------------|---------------------|--------------------|
|  | Face-2             | DV-Routing         | Face-2              | DV-Routing         |
| Gefundene Pfade                              | 95 %               | 99 %               | 86 %                | 97%                |
| Länge der gefundenen Pfade                   | 7,5 Hops<br>1385 m | 7,1 Hops<br>1410 m | 10,2 Hops<br>1490 m | 8,8 Hops<br>1525 m |
| Davon im Überbrückungsmodus                  | 0,85 Hops<br>32 m  | 0,30 Hops<br>49 m  | 2,45 Hops<br>85 m   | 0,85 Hops<br>104 m |
| Länge der nicht gefundenen Pfade             | 20 Hops<br>1940 m  | 8,1 Hops<br>1680 m | 20 Hops<br>1640 m   | 7,9 Hops<br>1310 m |
| Davon im Überbrückungsmodus                  | 15 Hops<br>930 m   | 3 Hops<br>620 m    | 15 Hops<br>720 m    | 3 Hops<br>450 m    |
| Datenvolumen für das erste Paket (1000 Byte) | 8,8 kB             | 11,2 kB            | 12,4 kB             | 19,8 kB            |
| Datenvolumen für jedes weitere Paket         | 8,8 kB             | 7,6 kB             | 12,4 kB             | 9,4 kB             |
| Überbrückungen pro gefundenem Pfad           | 0,15               | 0,20               | 0,46                | 0,59               |
| Hops pro erfolgreicher Überbrückung          | 5,5 Hops           | 1,6 Hops           | 5,4 Hops            | 1,5 Hops           |
| Paketanzahl pro erfolgreicher Überbrückung   | 5,5                | 375                | 5,5                 | 375                |

dings durch eine Erhöhung der Knotendichte nicht überwunden werden. Bei der zugrunde liegenden hohen Konnektivität hat die Knotendichte nahezu kein Einfluss auf die Länge der gefundenen Pfade; sie kann allerdings geringfügig die Anzahl der benötigten Überbrückungen reduzieren. Eine Erhöhung der Funkreichweite über 300 m hinaus beeinflusst weder die Anzahl der gefundenen Pfade noch die Anzahl der Überbrückungen oder deren Länge. Allerdings sinkt hierbei – aufgrund der zahlreichen halbwegs geraden Straßen – die durchschnittliche Pfadlänge des Basisverfahrens.

In der freien Ebene ist die durchschnittliche Knotenanzahl innerhalb der Funkreichweite – eine Gleichverteilung der Knoten vorausgesetzt – ausschlaggebend für die Konnektivität im Netzwerk. Diese Kenngröße wird sowohl durch die Funkreichweite als auch durch die Knotendichte beeinflusst. Beide Faktoren können dabei eine Veränderung des jeweils anderen Faktors ausgleichen. In der Innenstadt ist dies hingegen aufgrund der Funkhindernisse nicht ohne Weiteres möglich. Nur wenn sich mindestens ein Knoten in der unmittelbaren Nähe einer Straßenkreuzung befindet, können die Knoten auf den kreuzenden Straßen miteinander kommunizieren. Somit ist die Knotendichte im Vergleich zu der Funkreichweite die bedeutendere Einflussgröße.

Mit steigender Kommunikationsdistanz – also der Länge des kürzesten Pfads zwischen beiden Kommunikationspartnern – sinkt die Wahrscheinlichkeit, einen Pfad zu finden. Offensichtlich steigt gleichzeitig die durchschnittliche Länge der gefundenen Pfade an. Die Anzahl der be-

**Tabelle 3.4:** Einfluss von Knotendichte, Funkreichweite und Kommunikationsdistanz auf die GRANT-Varianten mit Überbrückungsverfahren.

|                           | Einfluss steigender Knotendichte | Einfluss steigender Funkreichweite | Einfluss steigender Kommunikationsdistanz |
|---------------------------|----------------------------------|------------------------------------|---|
| Gefundene Pfade           | ⊕                                | ⊖                                  | ⊖   |
| Pfadlänge [Hops]          | ⊖                                | ⊖                                  | ⊕   |
| Überbrückungen:           | ⊖                                | ⊖                                  | ⊕   |
| Überbrückungslänge [Hops] | ⊖                                | ⊖                                  | ⊖   |

nötigten Überbrückungen erhöht sich ebenfalls mit der Kommunikationsdistanz, wohingegen die durchschnittliche Länge der Überbrückungen weitgehend stabil bleibt.

Der Einfluss der bisher besprochenen Faktoren auf die Anzahl der erzeugten Pakete und das verursachte Datenvolumen unterscheidet sich für beide Überbrückungsverfahren. Aus diesem Grund wurden dieser Aspekt sowie alle anderen signifikanten Unterschiede der beiden Verfahren in Tabelle 3.5 separat zusammengefasst. Die Bewertung der Verfahren bezieht sich dabei immer auf die Kombination mit GRANT und soll in erster Linie dazu dienen, die jeweilige Eignung des Verfahrens abschätzen zu können.



























Sowohl die Knotendichte als auch die Funkreichweite erhöhen die im Durchschnitt zu erwartende Paketanzahl pro flutbasierter Überbrückungen deutlich, wenngleich Letztere diese Wirkung deutlich schwächer zeigt. Da allerdings das Basisverfahren bei höherer Knotendichte mehr Pfade findet und somit seltener eine Überbrückung benötigt, bleibt das Datenvolumen pro Zustellungsversuch bei beiden untersuchten Knotendichten in etwa gleich. Bei Face-2 sinkt durch diesen Effekt das Datenvolumen sogar geringfügig. Mit steigender Funkreichweite wiederum sinkt die durchschnittliche Länge der gefunden Pfade, wodurch das zusätzliche Datenvolumen aufgrund der höheren Paketanzahl pro flutbasiertem Suchvorgang in etwa ausgeglichen wird.

Der starke Einfluss von Knotendichte und Funkreichweite auf die Anzahl der Pakete, die bei einer flutbasierten Suche erzeugt werden, erschwert die Wahl des verwendeten Suchradius erheblich. Im Unterschied dazu hat die Wahl eines suboptimalen Limits der Überbrückungslänge bei Face-2 deutlich schwächere Auswirkungen. Zwar erhöht sich die zurückgelegte Strecke bis zum Abbruch einer aufgrund der Planarisierungsfehler entstandenen Endlosschleife, aber der Einfluss auf die durchschnittlichen Übertragungskosten ist weit geringer. Andererseits kann bei DV-Routing in Umgebungen mit niedriger Konnektivität des Netzwerks durch eine Erhöhung des Suchradius die Wahrscheinlichkeit, einen Pfad zu finden, erhöht werden, wohingegen bei Face-2 aufgrund der Planarisierungsfehler die Anzahl der maximal auffindbaren Pfade von vorneherein limitiert ist.

Als Entscheidungshilfe wird in der Tabelle auch die jeweilige Eignung für bestimmte Szenarien bewertet. Wenn beispielsweise viele Kommunikationspartner gleichzeitig Daten austauschen, werden durch die flutbasierten Suchvorgänge auf vielen Knoten Routingtabellen aufge-

### 3.5 Vergleich der Überbrückungsverfahren

**Tabelle 3.5:** Eigenschaften der beiden Überbrückungsverfahren in Kombination mit GRANT.

|  | GRANT-Face2   | GRANT-DV  |
|--|---|---|
| Funktionsprinzip   | Positionsbasiert  | Topologiebasiert  |
| Robust gegenüber einer zu hohen Limitierung der Überbrückungslänge |    |    |
| Kann jeden existierenden Pfad finden (bei steigenden Kosten)       |    |    |
| Vermeidet Fluten (bei bekannter Zielposition)                      |    |    |
| Vermeidet Endlosschleifen (bei gleichbleibender Topologie)         |    |    |
| <b>Einfluss steigender Knotendichte</b>                            |   |   |
| Paketanzahl pro Überbrückung                                       |    |    |
| Datenvolumen pro Zustellungsversuch                                |    |    |
| <b>Einfluss steigender Funkreichweite</b>                          |   |   |
| Paketanzahl pro Überbrückung                                       |   |   |
| Datenvolumen pro Zustellungsversuch                                |  |  |
| <b>Einfluss steigender Kommunikationsdistanz</b>                   |   |   |
| Paketanzahl pro Überbrückung                                       |  |  |
| Datenvolumen pro Zustellungsversuch                                | Proportional  | Proportional  |
| <b>Eignung für bestimmte Szenarien</b>                             |   |   |
| Viele gleichzeitig aktive Kommunikationspartner                    |  |  |
| Hohe Mobilität der Knoten  |  |  |
| Geringes, gelegentlich anfallendes Datenaufkommen                  |  |  |
| Hohes, zusammenhängendes Datenaufkommen                            |  |  |
| <b>Verbesserungsmöglichkeiten</b>                                  |   |   |
|  | Speichern von Abkürzungen   | Schrittweise Erhöhung des Suchradius  |
|  | Reduzierte Übertragungsleistung während der Überbrückung                            | Die Knotendichte berücksichtigender Flutmechanismus                                   |
|  |   | Kombination von Flutmechanismus und Lokalisierung                                     |

baut, die in anderen Überbrückungsfällen direkt verwendet werden können. In solchen Fällen kann auf die flutbasierten Suche verzichtet werden, sodass die Belastung des Netzwerkes abnimmt. Ein anderer wichtiger Aspekt ist die Knotenbewegung. Je höher die Knotenmobilität ist, desto kürzer sind im Durchschnitt die von DV-Routing gefundenen Überbrückungspfade gültig. Dementsprechend wird bei einer starken Knotenbewegung in vielen Fällen ein Reparaturmechanismus benötigt, der letztlich auch nur durch Fluten eine neue Überbrückung finden kann.

Bei einer moderaten Netzwerkauslastung dürfte in den meisten Fällen die Paketweiterleitung während einer Face-2-Überbrückung so wenig Zeit benötigen, dass sich die Knoten währenddessen nur minimal bewegen. Allerdings sind die Positionen der jeweiligen Nachbarn nur bekannt, wenn ein regelmäßiger Nachrichtenaustausch durchgeführt wird, der normalerweise nicht synchronisiert erfolgt. Somit sind die während der Weiterleiteraushwahl von Face-2 verwendeten Positionsinformationen mit hoher Wahrscheinlichkeit unterschiedlich aktuell. Würden alle während einer Überbrückung besuchten Knoten die identischen Positionsinformationen verwenden, wäre das unterschiedliche Alter der Informationen nur wenig bedeutend. Aufgrund möglicher Übertragungsfehler und des nicht synchronisierten Positionsaustauschs ist in einer realen Umgebung allerdings davon auszugehen, dass sich die verwendeten Standorte auf den einzelnen Knoten unterscheiden. Des Weiteren hängt die Genauigkeit, mit der ein Knoten seine eigene Position bestimmen kann, von dem verwendeten Positionierungssystem ab. Die konkrete Abweichung zwischen tatsächlicher und gemessener Position ist im Verhältnis zu der Funkreichweite meist unbedeutend. Allerdings können die Ungenauigkeiten in den gemessenen Standorten – selbst wenn sich der tatsächliche Aufenthaltsort nicht ändert – zu deutlichen Positionsunterschieden auf den an einer Überbrückung beteiligten Knoten führen und somit den beschriebenen Effekt der unterschiedlichen Aktualität der Informationen verstärken.

Für die Greedy-Verfahren sind die in diesem Zusammenhang zu erwartenden Ungenauigkeiten eher unbedeutend. Falls sich ein Knoten real drei Meter von der vermuteten Position entfernt befindet, verändert sich beispielsweise sein Abstand zum Zielknoten um maximal drei Meter. GRS würde in einem solchen Fall möglicherweise nicht den Knoten auswählen, der dem Ziel tatsächlich am nächsten ist, sondern einen anderen, der geringfügig weiter entfernt ist. Bei einer Funkreichweite von mehreren hundert Metern wären die durchschnittlichen Auswirkungen der suboptimalen Weiterleiteraushwahl allerdings äußerst gering. Selbst wenn sich der Knoten aus der Funkreichweite entfernt haben sollte, würde dies bei einem Einsatz von IEEE 802.11 durch die fehlende Bestätigung des Unicast-Pakets bemerkt werden.

Während der Validierungsphase des Simulators wurde unter anderem auch die Implementierung von Face-2 gründlich getestet. Beispielsweise wurde überprüft, ob Face-2 in der freien Ebene als alleiniges Routingverfahren für jeden Knoten einen Pfad zu jedem anderen Knoten finden kann. Trotz mehrfacher Überprüfung des Algorithmus wurde nur für einen Bruchteil der Kommunikationspartner ein Pfad gefunden. Die Ursache hierfür konnte nach vielen Tests darauf zurückgeführt werden, dass bei den Winkelberechnungen 32-Bit-Gleitkommazahlen verwendet wurden und es hierbei zu minimalen Rundungsfehlern gekommen ist, die in vielen

Fällen tatsächlich zu einer fehlerhaften Weiterleiteraushwahl geführt haben. Nachdem die Berechnung auf 64-Bit-Gleitkommazahlen umgestellt worden war, funktionierte die Implementierung wie erwartet. Die Ursache für die fatalen Folgen einer falschen Weiterleiteraushwahl aufgrund eines Rundungsfehlers weit hinter der vierten Kommastelle der Winkelberechnung liegt darin, dass Face-2 im Unterschied zu den Basisverfahren nicht ausschließlich die lokalen Informationen des jeweiligen Knotens verwendet, sondern implizit auch die topologischen Eigenschaften des planarisierten Netzwerkgraphen nutzt. Dadurch führt eine fehlerhafte Auswahl des nächsten Weiterleiters mit hoher Wahrscheinlichkeit zu weiteren fehlerhaften Entscheidungen. Neben dieser Problematik könnten die Positionsunterschiede auch die ohnehin existierenden Planarisierungsfehler verstärken, sodass die Anzahl der gefundenen Pfade signifikant sinken würde. Auch wenn in dieser Arbeit der Einfluss von Positionsungenauigkeiten nicht per Simulation untersucht wird, so kann aus den beschriebenen Gründen doch davon ausgegangen werden, dass Face-2 von allen untersuchten Verfahren die höchsten Performanceeinbrüche erleiden würde.

Falls nur geringe Datenmengen in größeren Zeitintervallen übertragen werden, muss DV-Routing jedes Mal erneut auf die flutbasierte Suche zurückgreifen. Das dabei erzeugte Datenvolumen verteilt sich auf nur wenige Datenpakete beziehungsweise wenige Nutzdaten, sodass die durchschnittlichen Übertragungskosten vergleichsweise hoch sind. Im Unterschied dazu ist das von Face-2 erzeugte Datenvolumen vorwiegend von der jeweiligen Topologie abhängig und wird nicht durch die Senderate beeinflusst. Folgerichtig sind bei der Übertragung von großen Datenmengen innerhalb kurzer Zeit die Suchkosten von DV-Routing vernachlässigbar, sodass die Übertragungskosten im Durchschnitt signifikant niedriger sind als bei Face-2.

Welches Überbrückungsverfahren besser geeignet ist, kann somit nicht universell beantwortet werden. Die Empfindlichkeit von Face-2 gegenüber Positionsungenauigkeiten und die Tatsache, dass aufgrund von Planarisierungsfehlern nicht für alle Kommunikationspartner ein Pfad gefunden werden kann, sprechen in vielen Einsatzszenarien gegen dieses Verfahren. Ein weiterer Nachteil sind die teilweise sehr langen Überbrückungen. Diese könnten verkürzt werden, indem beispielsweise einmal entdeckte Überbrückungspfade in den beteiligten Knoten gespeichert werden, sodass spätere Pakete die hinterlegten Pfade als Abkürzung verwenden könnten. Eine andere Verbesserungsmöglichkeit von Face-2 besteht darin, die Übertragungsleistung zum Weiterleiten auf dem planarisierten Graphen zu reduzieren, um so die Beeinträchtigung der benachbarten Knoten zu verringern.

Ein wesentlicher Vorteil von DV-Routing ist die Möglichkeit, durch einen großen Suchradius mit prinzipiell jedem Knoten kommunizieren zu können. Allerdings sollte aufgrund der stark wachsenden Paketanzahl der Suchradius – wie auch bei AODV – nur schrittweise erhöht werden. Die empfohlene Limitierung der Überbrückungslänge auf drei Hops könnte dabei als Startwert verwendet werden.

Eine Verbesserungsmöglichkeit von DV-Routing besteht darin, den Flutmechanismus auf die Knotenmenge innerhalb der Funkreichweite anzupassen. Beim Abschätzen der Kosten wird davon ausgegangen, dass jeder Knoten im Suchradius, der keinen entsprechenden Pfad kennt,

das Anfragepaket exakt einmal wiederholt. Diese Strategie entspricht dem in [71] beschriebenen Verfahren und ist vergleichsweise simpel. Zwar existieren effizientere Ansätze, bei denen wesentlich weniger Pakete übertragen werden, allerdings tauschen diese auch kontinuierlich zusätzliche Informationen aus und sind somit für eine sporadisch ausgeführte Überbrückungssuche ungeeignet. In [116] wird ein Verfahren vorgeschlagen, bei dem ein empfangenes Broadcast-Paket nur wiederholt wird, wenn nach einer zufälligen Wartezeit nicht schon mehr als  $c$  Kopien des Pakets empfangen wurden. In vielen Einsatzszenarien ist es wichtig, dass bei einem großflächigen Broadcast jeder Knoten eine Kopie des verschickten Pakets erhält. In Anbetracht der hohen Kosten kann bei der Suche nach einem Überbrückungspfad allerdings durchaus hingenommen werden, dass möglicherweise wenige Knoten die Anfrage nicht erhalten. Deshalb könnte in Einsatzgebieten mit einer hohen Knotendichte ein entsprechend geeignetes  $c$  gewählt oder anhand der Anzahl der Nachbarn geschätzt werden. Eine Übersicht über gängige Broadcast-Verfahren für drahtlose Ad-hoc-Netzwerke inklusive Simulationsstudie findet sich in [159].

Sollte die Position des Ziels nicht a priori bekannt sein, muss diese mit Hilfe eines Lokalisierungsservices ermittelt werden, der die gewünschte Position letztlich nur durch zusätzlichen Datenaustausch bestimmen kann. Positionsbasierten Greedy-Verfahren liegt die Annahme zugrunde, dass in hoch mobilen Umgebungen die aktuell ermittelte Position sicherer zum Ziel führt als ein explizit aufgebauter und gespeicherter Pfad. Da die Überbrückungen allerdings deutlich kürzer sind als die jeweiligen vollständigen Pfade, könnte der Datenaustausch zumindest für das Speichern von kurzen Pfadabschnitten genutzt werden, auf welche im Überbrückungsfall zurückgegriffen werden kann.

Aufgrund der besprochenen Eigenschaften ist DV-Routing für die meisten Einsatzszenarien als Überbrückungsstrategie besser geeignet als Face-2. Da Verfahren mit Distanzvektoren weitverbreitet und dementsprechend gut erforscht sind, ist diese Kombination eine gute Ausgangsbasis für ein hybrides Protokoll, welches durch die vorgeschlagenen Verbesserungsmöglichkeiten auch in Umgebungen mit einer hohen Knotendichte gute Ergebnisse erzielt. Des Weiteren bietet diese Kombination die Möglichkeit, dass DV-Routing notfalls den gesamten Pfad überbrücken kann, sodass auch eine Kommunikation mit Knoten möglich wäre, die ihre eigene Position nicht bestimmen können.

### 3.6 Zusammenfassung

Zu Beginn dieses Kapitels wurden die üblicherweise für die Bewertung von Ad-hoc-Routingprotokollen verwendeten Kriterien besprochen. Dabei wurde deutlich, dass die verwendeten Simulationsmodelle nur unzureichend für die Beurteilung der Leistungsfähigkeit von positionsbasierten Routingheuristiken geeignet sind. Aus diesem Grund wurde ein neuer Simulator vorgestellt, der anhand von Momentaufnahmen großflächige Analysen ermöglicht. Ein wichtiges Ergebnis der vorgestellten Simulationsstudie ist die Erkenntnis, dass die positionsbasierten Ansätze in einem Netzwerk, in dem alle Knoten gleichmäßig in der Ebene ver-

teilt sind, nahezu immer einen Pfad finden können. Sobald die Knoten jedoch ungleichmäßig verteilt sind und Funkhindernisse berücksichtigt werden, brechen die Ergebnisse dramatisch ein. Daher wurden sowohl mehrere Verbesserungsmöglichkeiten des Basisverfahrens als auch die Ergänzung mit einer Überbrückungsstrategie besprochen und evaluiert. Aufgrund der unbefriedigenden Ergebnisse und der hohen Überbrückungskosten wurde mit GRANT ein neuer positionsbasierter Routingansatz vorgestellt, der signifikant mehr Pfade findet als GRS und daher deutlich seltener auf eine Überbrückung angewiesen ist. Abschließend wurden die beiden untersuchten Überbrückungsstrategien im Detail evaluiert.





## Kapitel 4

# Probabilistische Datenstrukturen für kooperative Anwendungen

Übe dich auch an den Dingen,  
an denen du verzweifelst.

---

– Marc Aurel –

Die in den vorherigen Abschnitten beschriebenen Routingverfahren können eingesetzt werden, um in einem Ad-hoc-Netzwerk zwischen zwei Kommunikationspartnern einen Pfad zu finden und über diesen Daten zu transportieren. Diese Funktionalität entspricht der Vermittlungsschicht im *OSI-Schichtenmodell* [8] beziehungsweise der Internetschicht im *TCP/IP-Referenzmodell* [29]. Als Bitübertragungs- und Sicherungsschicht beziehungsweise als Netzzugangsschicht kann in einem drahtloses Netzwerk beispielsweise IEEE 802.11 *Wireless Local Area Network* (WLAN) [80] eingesetzt werden. Im Unterschied zum *Internet Protocol* (IP) sind die besprochenen Routingverfahren speziell für eine drahtlose und dezentral organisierte Vernetzung ausgelegt. Unter Berücksichtigung der relevanten Schnittstellen zwischen den Netzwerkschichten kann IP prinzipiell vollständig durch eines der präsentierten Verfahren ersetzt werden. Alternativ besteht die Möglichkeit, die IP-Pakete als Nutzdaten in die Pakete des Ad-hoc-Routingprotokolls zu verpacken, zum Zielrechner weiterzuleiten, dort wieder zu entpacken und dem lokalen IP-Stack zu übergeben. Auf diese Weise können die im Internet verwendeten Transportprotokolle *User Datagram Protocol* (UDP) [124] und *Transmission Control Protocol* (TCP) [125] auch in einem Ad-hoc-Netzwerk eingesetzt werden.

Gängige TCP-Implementierungen interpretieren jeden Paketverlust als Indiz für eine Netzwerküberlastung. In einem drahtlosen Netzwerk können aber häufig Pakete nicht zugestellt werden, weil sich der Zielknoten aus der Funkreichweite des Senders entfernt hat. Des Weiteren ist die Wahrscheinlichkeit für Paketskollisionen oder andere Störeffekte viel höher als in einem drahtgebundenen Netzwerk, sodass TCP nur bedingt für den Einsatz in einem drahtlosen Ad-hoc-Netzwerk geeignet ist [20, 72]. Doch trotz dieser Limitierung bietet die Verwendung etablierter Transportprotokolle den Vorteil, dass nahezu alle auf diesen Protokollen aufbauenden Internetanwendungen ohne Anpassungen auch in einem drahtlosen Ad-hoc-Netzwerk verwendet werden können.

Wenngleich durch die speziellen Routingverfahren nahezu alle Internetanwendungen auch in einem drahtlosen Ad-hoc-Netzwerk verwendet werden können, stellt sich doch die Frage, ob die Netzwerkarchitektur tatsächlich für derartige Anwendungen geeignet ist. Sei  $n$  die Anzahl der Knoten des Ad-hoc-Netzwerks und  $W$  der maximal nutzbare Datendurchsatz bei einer konstanten Funkreichweite. Unter der Voraussetzung, dass jeder Knoten einen zufälligen Zielknoten auswählt, steht jedem Knoten eine Bandbreite von  $\Theta(W/\sqrt{n \log n})$  [67] zur Verfügung. Dementsprechend muss entweder das Datenaufkommen deutlich geringer sein als in drahtgebunden Netzwerken, oder aber die Pfadlänge muss strikt auf wenige Hops begrenzt werden. Des Weiteren sind inzwischen auch kostengünstige Internetzugänge über das Mobilfunknetz verfügbar. Dadurch ist es mobilen Geräten wie Smartphones oder Laptops möglich, in nur einem Hop mit den bereitgestellten Basisstationen zu kommunizieren und über diese Zugriff auf das Internet zu erhalten. Auf diese Weise können mobile Geräte über das Internet miteinander kommunizieren, unabhängig davon, wie weit sie voneinander entfernt sind. In Bezug auf die Topologie und die damit verbundene Skalierbarkeit ist dieser Ansatz einem drahtlosen Ad-hoc-Netzwerk deutlich überlegen und insbesondere schon heute flächendeckend verfügbar. Andererseits bieten Ad-hoc-Netzwerke den Vorteil, auch in Katastrophenfällen oder in Gebieten ohne Strom und Netzwerkinfrastruktur zu funktionieren. Besonders in solchen Einsatzgebieten ist es nützlich, bestehende Anwendungen ohne grundlegende Änderungen verwenden zu können.

Neben den Bemühungen, ein universell einsetzbares Ad-hoc-Netzwerk nach dem Vorbild des Internets zu formen, gibt es auch Ansätze, die ein Funknetzwerk primär als zweckgebundene Einheit betrachten. Zweckgebunden bedeutet in diesen Zusammenhang, dass das gesamte Netzwerk nur für das Lösen einer konkreten Aufgabenstellung bereitgestellt wird. Wenn beispielsweise ein Naturschutzgebiet überwacht werden soll, können sogenannte Sensorknoten – mit Funktechnik und Sensoren ausgestattete Kleinstcomputer – in dem Areal aufgestellt werden. Diese sammeln dezentral die gewünschten Daten, zum Beispiel Wärme- und Bewegungsinformationen, und leiten sie an die Datensinken weiter. Dabei werden meist dieselben Strategien wie in den beschriebenen Routingprotokollen angewendet. Der Unterschied zu diesen besteht darin, dass das gesamte Sensornetzwerk geschlossen ist und nur den spezifizierten Zwecken dient, sodass die Weiterleitung nicht zwangsläufig als getrennte und universell einsetzbare Netzwerkschicht gestaltet werden muss, sondern auch direkt mit der Anwendungsschicht verwoben werden kann. Gemäß dem Schichtenmodell ist der Aufbau der transportierten Nutzdaten unbekannt, sodass die Daten ohne Änderungen vom Absender zum Zielknoten weitergeleitet werden. Da aber in dem genannten Beispiel auf den Sensorknoten nur eine Anwendung vorhanden ist und die Schichten deshalb nicht zwangsläufig voneinander getrennt werden müssen, können die Daten während der Weiterleitung interpretiert und sogar verändert werden.

In diesem Beispiel wird die Leistung, die Sensordaten zu erfassen und zu übertragen, kooperativ von den Netzwerkknoten für den Nutzer beziehungsweise Besitzer des Netzwerks erbracht. In einem Netzwerk mit strikter Schichtentrennung wird hingegen die Funkionali-

---

tät der tieferen Netzwerkschichten kooperativ von den einzelnen Netzknoten erbracht und den höher liegenden Schichten bereitgestellt. Ein Softwareentwickler kann so auf standardisierte Schnittstellen zurückgreifen, um Daten zwischen verschiedenen Softwareinstanzen auszutauschen. In einem solchen Szenario muss auf jedem Knoten des Netzwerks ein kompatibles Routingprotokoll ausgeführt werden, welches die Nutzdaten der Anwendung inklusive der Informationen des Transportprotokolls auf dem Absendersystem verpackt, zum Zielsystem weiterleitet und dort wieder entpackt. Die eigentliche Software auf der Anwendungsebene muss dementsprechend nur auf den Systemen der beiden Kommunikationspartner verfügbar sein. Somit wird zwar die Weiterleitung der Pakete von den einzelnen Knoten kooperativ organisiert, die Kommunikation auf der Anwendungsebene aber erfolgt nur zwischen zwei – meist nicht direkt benachbarten – Knoten.

Kooperative Anwendungen sind von besonderem Interesse, weil sie den Rahmenbedingungen eines drahtlosen Ad-hoc-Netzwerks deutlich besser entsprechen als Anwendungen, die vorwiegend Ende-zu-Ende-Verbindungen verwenden. In diesem Zusammenhang ist es vor allem wichtig, dass alle Knoten innerhalb der Funkreichweite des Senders dessen Pakete ohne Weiteres empfangen können. Neben der expliziten Adressierung eines Zielknotens hat in einem drahtlosen Netzwerk somit jedes Paket auch eine implizite räumliche Adressierung. Wollen beispielsweise mehrere mit Funktechnik ausgestattete Fahrzeuge Verkehrsinformationen austauschen und aggregieren, dann ist es nicht sinnvoll, ein mehrere Straßen entferntes Fahrzeug explizit zu adressieren und nach den neuesten Daten zu fragen. Denn zum einen können die vermittelnden Knoten auch von den Daten profitieren, zum anderen besteht zu dem weiter entfernten Fahrzeug überhaupt kein Bezug, sodass auch dessen Adresse und Aufenthaltsort nicht a priori bekannt sind.

Neben der Kommunikationsform ist auch das verwendete Datenformat von hoher Bedeutung. Sollen beispielsweise mehrere einfache Basisbeobachtungen wie beispielsweise die Anzahl der verfügbaren Parkplätze oder die gemessene Temperatur miteinander kombiniert werden, so muss neben den eigentlichen Daten auch die Information übertragen werden, welcher Knoten die jeweiligen Daten ermittelt hat. Je nach Anwendung wird zusätzlich der Beobachtungszeitpunkt benötigt. Dadurch ergibt sich schon bei einer niedrigen Knotendichte ein sehr hohes Datenaufkommen. Wie bereits in Kapitel 2 erläutert, besteht ein Lösungsansatz für dieses Problem darin, anstelle der eigentlichen Daten eine sogenannte probabilistische Datenstruktur zu übertragen. In dieser werden nur gewisse statistische Merkmale der Daten gespeichert, sodass der benötigte Speicherplatz im Vergleich zu dem für die eigentlichen Daten benötigten enorm reduziert wird. Die in der Datenstruktur hinterlegten Merkmale können von den Empfängern genutzt werden, um die ursprünglichen Eingabedaten zu schätzen. Wie eine derartige Datenstruktur für die dezentrale Aggregation verwendet werden kann, wird in Abschnitt 4.2 genau beschrieben. Zuvor wird aber im nächsten Abschnitt eine für diesen Zweck geeignete probabilistische Datenstruktur im Detail analysiert und evaluiert.

## 4.1 Flajolet-Martin Sketches

Die Kardinalität (Mächtigkeit) einer endlichen Menge kann durch das einfache Abzählen ihrer Elemente bestimmt werden. Allerdings ist im IT-Umfeld häufig statt der eigentlichen Menge nur eine Datenansammlung bekannt, die auch Duplikate enthalten kann. Soll beispielsweise die Anzahl der unterschiedlichen Wörter in einem Text bestimmt werden, so ist die Menge der verwendeten Wörter zuvor unbekannt und muss während der Berechnung erst bestimmt werden. Eine einfache Lösungsmöglichkeit für diese Schwierigkeit besteht darin, die Datenansammlung sequenziell zu bearbeiten und dabei eine Liste der unterschiedlichen Wörter anzulegen, die im Anschluss durchgezählt werden können. Doch sowohl die Berechnungskomplexität, die notwendig ist, um festzustellen, ob das aktuell betrachtete Wort schon in der Liste enthalten ist, als auch der hierfür benötigte Speicher limitieren diesen Ansatz. Auch wenn dieses simple Verfahren noch deutlich verbessert werden kann, ist offensichtlich, dass der benötigte Speicherplatz unmittelbar von der Kardinalität der Menge abhängt. Falls eine näherungsweise Lösung ausreichend ist, kann der verwendete Speicherplatz durch den Einsatz von *Flajolet-Martin Sketches* (FM-Sketches) [55] drastisch gesenkt werden.

### 4.1.1 Aufbau und Eigenschaften

Ein FM-Sketch der Größe  $L > 0$  besteht aus einem Bitarray  $S = s_0, \dots, s_{L-1}$ , welches zu Beginn an allen Positionen mit 0 initialisiert wird. Sei  $hash(x)$  eine Hashfunktion, deren berechnete Hashwerte näherungsweise gleich verteilt sind. Sei des Weiteren  $l_b(x)$  eine Funktion, welche die Position des geringwertigsten Bits mit dem Wert  $b$  innerhalb der binären Repräsentation von  $x$  zurückgibt. Die Positionen werden dabei mit 0 beginnend durchgezählt. Sollte die binäre Repräsentation von  $x$  kein Bit mit dem Wert  $b$  enthalten, so gibt  $l_b(x)$  die Position des letzten Bits zurück. Ein Element  $x$  wird in den FM-Sketch  $S$  eingefügt, indem das Bit  $s_i$  mit  $i = l_1(hash(x))$  auf 1 gesetzt wird. Wenn beispielsweise die binäre Repräsentation des Hashwertes  $hash(„Test“)$  01110100 ist, dann befindet sich die erste 1 – mit 0 beginnend gezählt – an der ersten Position. Da dementsprechend  $l_1(01110100) = 1$  gilt, wird beim Einfügen das Bit  $s_1$  auf 1 gesetzt.

Da die Hashfunktion gleich verteilte Hashwerte berechnet, besteht an jeder Position der binären Repräsentation eine 50-prozentige Wahrscheinlichkeit, dass eine 1 beziehungsweise eine 0 generiert wird. Daraus ergibt sich folgende Gleichung:

$$P(l_1(hash(x)) = i) = 2^{-(i+1)}. \quad (4.1)$$

Somit entspricht die Wahrscheinlichkeit, an welcher Position im Sketch ein Element eingefügt wird, einer geometrischen Verteilung. Aus der beschriebenen Einfügeprozedur wird ersichtlich, dass die Datenstruktur nicht das Element selbst speichert, sondern nur ein statistisch auswertbares Merkmal dieses Elements. Konkret ist im Sketch an der Stelle  $i$  gespeichert, ob bisher ein Element mit einer Auftrittswahrscheinlichkeit von  $2^{-(i+1)}$  eingefügt wurde. Das

Speichern dieser Merkmale erfolgt somit sehr kompakt und ist unabhängig von der Anzahl der bisher eingefügten Elemente. Der Parameter  $L$  beeinflusst dabei sowohl den benötigten Speicherplatz eines Sketches als auch die maximale Anzahl der Elemente, die gezählt werden können.

Nachdem eine größere Anzahl von Elementen in den Sketch eingefügt wurde, beginnt die binäre Repräsentation von  $S$  mit mehreren Einsen. Bei  $S = 11100100$  stehen beispielsweise drei aufeinanderfolgende Einsen auf der linken Seite des Sketches. Da  $l_0(S)$  die Position der ersten 0 von links zurückgibt, kann diese Funktion zur Bestimmung der Länge der Sequenz verwendet werden. Auf dieser Kenngröße basierend kann die Anzahl der bisher eingefügten Elemente durch

$$C(S) = \frac{2^{l_0(S)}}{p} \quad (4.2)$$

mit  $p \approx 0,775351$  abgeschätzt werden. Gemäß ihrer Definition kann diese Abschätzung nur  $L$  unterschiedliche Werte annehmen, wobei die maximal geschätzte Anzahl  $2^L/p$  beträgt. Wenn beispielsweise  $L = 16$  gilt, ist folgerichtig die Länge der Sequenz auf 16 begrenzt. Dementsprechend beträgt die maximal geschätzte Anzahl von Elementen circa 84.524. Gleichzeitig kann die Abschätzung aber maximal 16 unterschiedliche Werte annehmen und ist somit sehr ungenau.

Deshalb schlagen die Autoren eine Technik mit dem Namen *Probabilistic Counting with Stochastic Averaging* (PCSA) vor. Dabei wird anstelle eines einzelnen Sketches eine Menge von  $M > 0$  Sketches eingesetzt, die im Folgenden mit  $S_0, \dots, S_{M-1}$  bezeichnet werden. Ein Element  $x$  wird in den Sketch  $S_j$  mit  $j = \text{hash}(x) \bmod M$  eingefügt, indem das Bit  $s_i$  mit  $i = l_1(\text{hash}(x)/M)$  auf eins gesetzt wird. Durch die Modulo-Operation werden die eingefügten Elemente annähernd gleich auf alle Sketches verteilt. Abgesehen davon, dass der Hashwert durch  $M$  dividiert wird, bleibt die Einfügeprozedur unverändert. Die Anzahl der unterschiedlichen Elemente, die bisher eingefügt wurden, kann mit

$$C(S_0, \dots, S_{M-1}) = \frac{M}{p} \cdot 2^{\sum_{j=0}^{M-1} l_0(S_j)/M} \quad (4.3)$$

abgeschätzt werden. Analog zu Formel 4.2 wird auch in Formel 4.3 mit Hilfe der Funktion  $l_0(S)$  die Länge der Folge von Einsen bestimmt. Doch statt anhand der jeweiligen Länge die Ergebnisse der Sketches einzeln zu berechnen, wird die Länge über alle Sketches gemittelt. Diese gemittelte Länge wird dann wie in Formel 4.2 verwendet, um die Anzahl der eingefügten Elemente abzuschätzen. Da es sich dann nur um das durchschnittliche Ergebnis pro Sketch handelt, wird dieses noch mit der Sketchanzahl  $M$  multipliziert. Unter der Voraussetzung, dass mehr als  $15 \cdot M$  Elemente eingefügt wurden, hat diese Funktion einen mittleren Schätzfehler von circa  $0,78/\sqrt{M}$ . Um bei nur wenig gefüllten Sketches die Genauigkeit zu verbessern, schlagen die Autoren von [134] folgende Modifikation vor:

$$C(S_0 \dots S_{M-1}) = \frac{M}{p} \cdot \left( 2^{\sum_{j=0}^{M-1} l_0(S_j)/M} - 2^{-1,75 \cdot \sum_{j=0}^{M-1} l_0(S_j)/M} \right). \quad (4.4)$$

Diese Abschätzung reduziert die anfängliche Ungenauigkeit und nähert sich mit steigendem Füllstand der Sketches asymptotisch der Formel 4.3 an.

Die Einfügeoperation ist deterministisch und unabhängig vom aktuellen Inhalt des Sketches. Falls ein Element mehrmals eingefügt wird, so wird jedes Mal das identische Bit auf eins gesetzt. Diese Eigenschaften erlauben es, dass mehrere Sketches mit einer bitweisen OR-Operation kombiniert werden können. Der so entstandene Sketch kann dazu verwendet werden, um die Summe der unterschiedlichen Elemente aller Sketches zu schätzen. Abbildung 4.1 zeigt die exemplarische Kombination der Sketches A und B.

|    |   |   |          |          |   |   |   |   |  |
|----|---|---|----------|----------|---|---|---|---|--|
|    | 0 | 1 | 2        | 3        | 4 | 5 | 6 | 7 |  |
| A  | 1 | 1 | 1        | <u>0</u> | 0 | 1 | 0 | 0 | $C(A) = \frac{2^{l_0(A)}}{p} \approx 10$ |
| OR |   |   |          |          |   |   |   |   |  |
| B  | 1 | 1 | <u>0</u> | 1        | 0 | 0 | 0 | 0 | $C(B) = \frac{2^{l_0(B)}}{p} \approx 5$  |
| =  |   |   |          |          |   |   |   |   |  |
| D  | 1 | 1 | 1        | <u>0</u> | 1 | 1 | 0 | 0 | $C(D) = \frac{2^{l_0(D)}}{p} \approx 21$ |

Abbildung 4.1: Kombination von zwei FM-Sketches durch eine bitweise OR-Operation.

Die erste 0 befindet sich in den Sketches A und B an der Position 3 beziehungsweise 2. Dementsprechend ergibt die Abschätzung gemäß Formel 4.2, dass etwa zehn beziehungsweise fünf unterschiedliche Elemente in den jeweiligen Sketch eingefügt wurden. Bei dem durch die bitweise OR-Operation entstandenen Sketch D ergibt die Abschätzung etwa 21 Elemente. Auch wenn in diesem Beispiel  $C(D)$  größer ist als  $C(A) + C(B)$ , ist die Kombination von Sketches doch insbesondere für die Bestimmung der Kardinalität von dezentral erfassten und mit Duplikaten versetzten Daten geeignet. Die Kombination dient somit keinesfalls dazu, die Abschätzung zu verbessern, sondern lediglich dazu, den benötigten Speicherplatz von Duplikaten zu befreien und zu reduzieren.

#### 4.1.2 Relativer Fehler und Standardabweichung

Da es sich bei der Anwendung von Sketches um ein probabilistisches Verfahren handelt, ist es von hohem Interesse, den zu erwartenden Schätzfehler zu kennen. Die Autoren leiten her, dass der mittlere Schätzfehler von PCSA circa  $0,78/\sqrt{M}$  beträgt, wobei  $M$  die Anzahl der parallel verwendeten Sketches bezeichnet. Dabei handelt es sich um die durchschnittlich zu erwartende relative Abweichung zwischen der realen und der geschätzten Kardinalität. Welche Genauigkeit benötigt wird, hängt von der einzelnen Anwendung ab und kann nicht generell

beantwortet werden. Zu berücksichtigen ist allerdings der Umstand, dass sich der benötigte Speicherbedarf mit steigender Genauigkeit sehr schnell erhöht. Bei  $M = 64$  kann von einem mittleren Schätzfehler von 9,8 % ausgegangen werden. Wird pro Sketch  $L = 16$  Bit verwendet, benötigen die 64 Sketches 128 Byte. Durch die Verdopplung von  $M$  kann der mittlere Schätzfehler auf 6,9 % gesenkt werden. Dementsprechend müsste der verwendete Speicherplatz deutlich erhöht werden, um den mittlere Schätzfehler auf wenige Prozent zu reduzieren.

Die Fehlerabschätzung wird in der Originalpublikation mathematisch hergeleitet und geht von einer idealisierten Hashfunktion aus, deren Hashwerte in der binären Repräsentation über die Bitpositionen gleich verteilt sind. Demgegenüber wird in diesem Abschnitt die Genauigkeit in einer realen Implementierung empirisch untersucht. Zwar wurden FM-Sketches von zahlreichen Publikationen aufgegriffen, die vorliegenden Evaluationen aber beziehen sich überwiegend auf den jeweiligen Kontext und beinhalten somit auch andere Einflussquellen wie beispielsweise Übertragungsfehler in einem Netzwerk. Deshalb wird in diesem Abschnitt empirisch untersucht, welche Genauigkeit mit einer etablierten Hashfunktion unabhängig von der konkreten Anwendung erreicht werden kann.

Der angegebene Standardfehler gilt nur, wenn zuvor mindestens  $15 \cdot M$  Elemente eingefügt wurden. Des Weiteren sollen den Autoren zufolge die letzten vier Bits der Sketches noch nicht durchgehend mit Einsen gefüllt sein. Ausgehend von einem Bitarray mit  $L = 16$  sollte die Folge von Einsen somit nicht länger als 12 sein, sodass ein Sketch unter optimalen Umständen bis maximal  $2^{12}/p \approx 5283$  zählen kann. Insbesondere wenn mehrere Sketches verwendet werden ist allerdings die Wahrscheinlichkeit hoch, dass zumindest bei einer Teilmenge die hinteren Bits schon wesentlich früher belegt werden. Aus diesem Grund wird in der folgenden Evaluation bei  $L = 16$  ein Limit von 4000 Elementen pro Sketch verwendet.

Das Evaluationsprogramm erzeugt eine Menge von  $M$  Sketches und initialisiert alle Bits mit Null. Anschließend werden schrittweise  $4000 \cdot M$  Elemente eingefügt. Nach jeder Einfügeoperation berechnet das Programm mit Hilfe der aktuellen Schätzung  $x_a$  und der tatsächlichen Anzahl der bereits eingefügten Elemente  $x_r$  den relativen Fehler

$$F_r = \frac{x_a - x_r}{x_r} \quad (x_r \neq 0). \quad (4.5)$$

Abhängig vom Ergebnis der Subtraktion im Zähler kann der relative Fehler sowohl negativ als auch positiv sein. Aus diesem Grund wird im Folgenden der systematische Fehler der Schätzung mit Hilfe des Mittelwerts von  $F_r$  beurteilt. Im Idealfall sollten sich die Unter- und Überschätzungen über den gesamten Wertebereich hinweg ausgleichen, was einem Mittelwert von exakt Null entsprechen würde. Des Weiteren wird der im Durchschnitt zu erwartende relative Fehler der Schätzung anhand des Mittelwerts von  $|F_r|$  berechnet. Um den Verlauf des relativen Fehlers über den Wertebereich zu untersuchen, wird zusätzlich der gesamte Wertebereich in Intervalle unterteilt, für die ebenfalls der Mittelwert und die Standardabweichung bestimmt werden.

In einer realen Anwendung ist der Aufbau der zu zählenden Elemente durch den jeweiligen Einsatz vorgegeben. Wenn beispielsweise die Anzahl der unterschiedlichen Wörter in einem



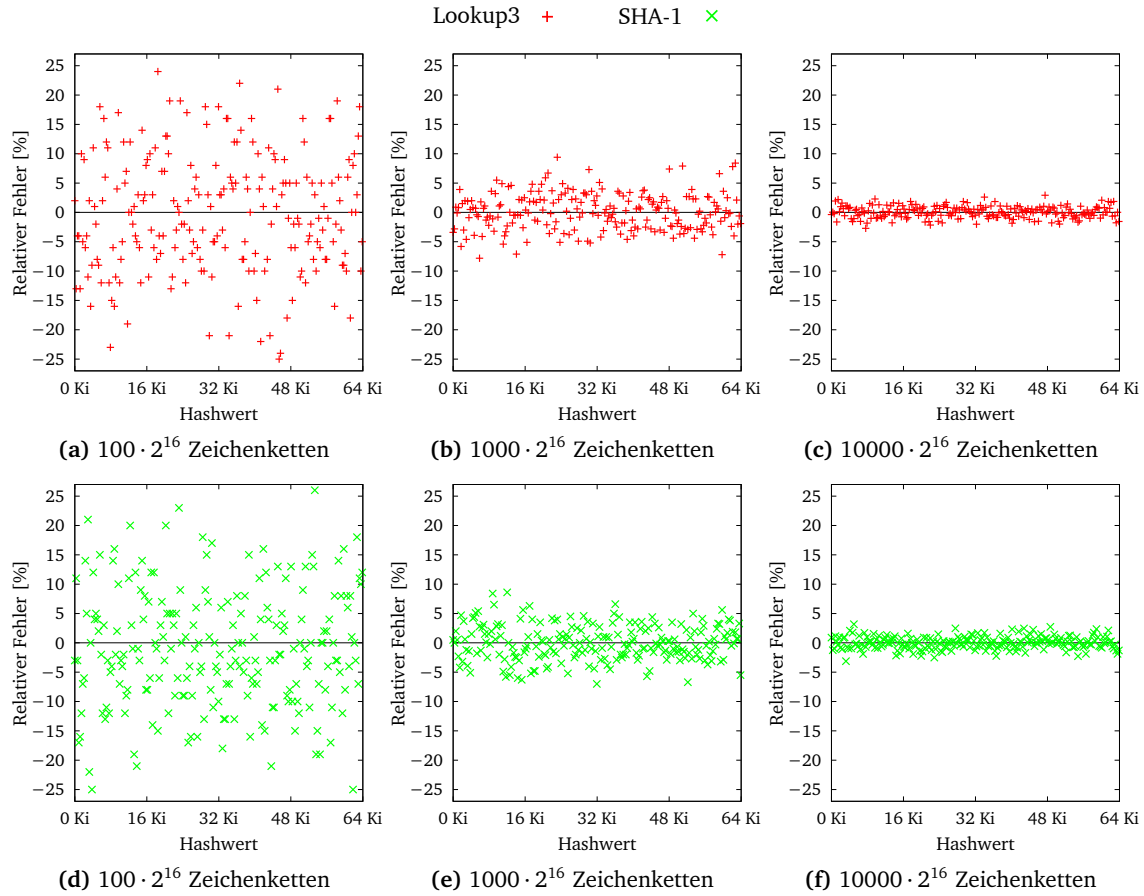
Text bestimmt werden soll, entspricht die Zeichenkette jedes Worts einem Element. Beim Einfügen werden diese Zeichenketten mit Hilfe der verwendeten Hashfunktion in Zahlen umgewandelt und ausgewertet. In der mathematisch hergeleiteten Fehlerabschätzung wird von einer idealisierten Hashfunktion ausgegangen, die gleich verteilte Hashwerte generiert. In einer realen Implementierung stellt sich die Frage, inwieweit gängige Hashfunktionen diese Annahme erfüllen. Dabei ist insbesondere von Interesse, ob die Gleichverteilung der Hashwerte auch bei ähnlichen Zeichenketten gewährleistet ist. Aus diesem Grund verwendet das Evaluationsprogramm zwei unterschiedliche Strategien, um Zeichenketten synthetisch zu generieren. In der ersten Strategie wird ausgehend von der Zahl  $x$  und dem Durchlauf  $r$  eine Zeichenkette der Form  $x:r:x$  erzeugt. Da  $r$  für  $4000 \cdot M$  Einfügeoperationen unverändert bleibt, soll durch die zweifache Verwendung von  $x$  sichergestellt werden, dass sich die Zeichenkette nicht nur an einer Stelle von der vorherigen unterscheidet. Um die Zeichenketten möglichst unterschiedlich zu gestalten, werden in der zweiten Strategie zusätzlich die Zufallszahlen  $a$  und  $b$  bestimmt. Um die Zeichenkette zu generieren, wird dann die Vorlage  $x:a:r:b:x$  verwendet.

Die Folge von Zahlen, die ein Zufallszahlengenerator berechnet, ist deterministisch, sodass mit Hilfe der sogenannten Saat zu jeder Position der Folge gesprungen werden kann. Wird die Saat für die Berechnung der Zufallszahlen direkt aus dem zu zählenden Element gewonnen, können zwei unterschiedliche Instanzen für das gleiche Element auch die gleiche Zeichenkette erzeugen. Beispielsweise kann für das Element  $x$  die Saat  $\text{hash}(x)$  verwendet werden. Somit ist diese Strategie nicht auf eine Simulationsumgebung beschränkt, sondern kann auch in verteilten Anwendungen eingesetzt werden, die FM-Sketches über das Netzwerk austauschen und miteinander kombinieren.

Das wichtigste Kriterium, um die Eignung einer Hashfunktion für den Einsatz innerhalb von PCSA zu bewerten, ist die Verteilung der von ihr generierten Hashwerte. Aus diesem Grund werden die Hashfunktionen *Lookup3* [82] und *Goulburn* [119] mit Hilfe der beiden Strategien zur Generierung von Zeichenketten vorab evaluiert. Des Weiteren werden auch die kryptografischen Hashfunktionen *SHA-1* [156] und *Skein-256* [50] untersucht. Um den Wertebereich etwas einzuschränken, werden nur die ersten 16 Bit der berechneten Hashwerte berücksichtigt, sodass alle Hashwerte zwischen 0 und  $2^{16}-1$  liegen. Das Programm berechnet für  $100 \cdot 2^{16}$  Zeichenketten die Hashwerte und zählt, wie oft jeder Hashwert auftritt. Anschließend wird für jede Zahl im Wertebereich der relative Fehler in Bezug auf die Abweichung der realen von der zu erwartenden Häufigkeit bei einer perfekten Gleichverteilung bestimmt. Diese Prozedur wird ebenfalls für 1000 beziehungsweise  $10000 \cdot 2^{16}$  Zeichenketten durchgeführt. Die Ergebnisse für *Lookup3* und *SHA-1* sind in Abbildung 4.2 zu sehen. Um die Übersichtlichkeit zu verbessern, wurde nur für jeden 300. Hashwert der relative Fehler eingezeichnet.

Die dargestellten Ergebnisse sind repräsentativ für alle vier untersuchten Hashfunktionen und für beide Strategien zur Zeichenkettengenerierung. Dementsprechend sind alle untersuchten Hashfunktionen in der Lage, auch ähnliche Zeichenketten annähernd gleich verteilt im jeweiligen Wertebereich abzubilden. Bezüglich der Geschwindigkeit der Konvergenz in Richtung einer Gleichverteilung unterscheiden sich die untersuchten Hashfunktionen ebenfalls nur ge-

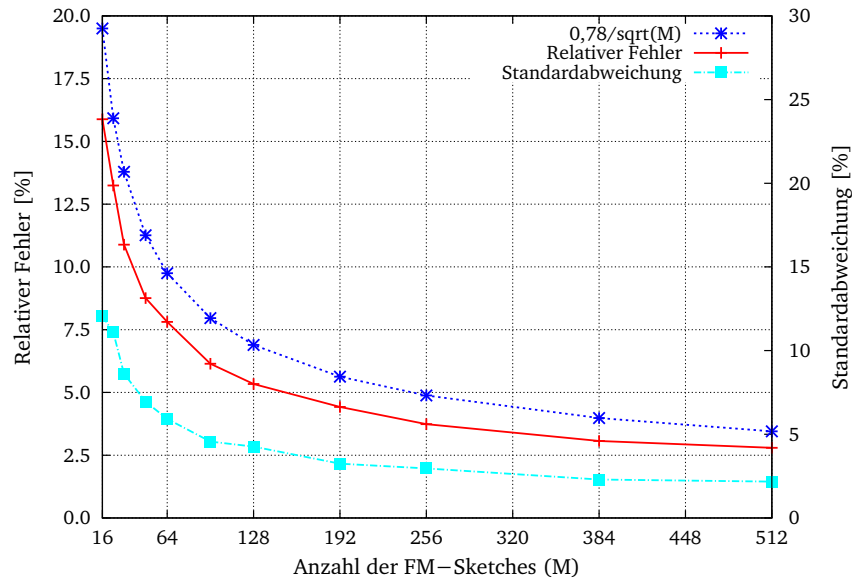




**Abbildung 4.2:** Die Verteilung der Hashwerte von Lookup3 und SHA-1. Der angegebene relative Fehler bezieht sich auf die Abweichung der realen von der zu erwartenden Häufigkeit bei einer perfekten Gleichverteilung.

ringfügig. Aus diesen Gründen wird in den folgenden Untersuchungen nur noch die Hashfunktion Lookup3 verwendet. Da die Zufallszahlen keinen signifikanten Einfluss auf die Ergebnisse hatten, werden im Folgenden alle Zeichenketten gemäß der ersten Strategie generiert.

Abbildung 4.3 zeigt den Mittelwert und die Standardabweichung von  $|F_r|$  in Abhängigkeit von der Anzahl der verwendeten FM-Sketches. Alle angegebenen Werte entsprechen dem jeweiligen Durchschnitt des gesamten Wertebereichs und von einhundert Simulationsdurchläufen. Zusätzlich wurde auch der mathematisch hergeleitete mittlere Schätzfehler  $0,78/\sqrt{M}$  eingezeichnet. Die präsentierten Ergebnisse zeigen, dass Letzterer in der realen Anwendung sogar deutlich unterboten werden kann. Bei 32 FM-Sketches beträgt der durchschnittliche relative Fehler circa 10,8 %. Diese Konfiguration benötigt bei  $L = 16$  insgesamt 64 Byte und kann ohne Genauigkeitsverlust mindestens  $32 \cdot 4000 = 128000$  Elemente zählen. Wird die Anzahl der FM-Sketches auf 128 gesteigert, vergrößern sich sowohl der Speicherbedarf als auch der Wertebereich proportional. Allerdings sinkt gleichzeitig der durchschnittliche relative Fehler auf

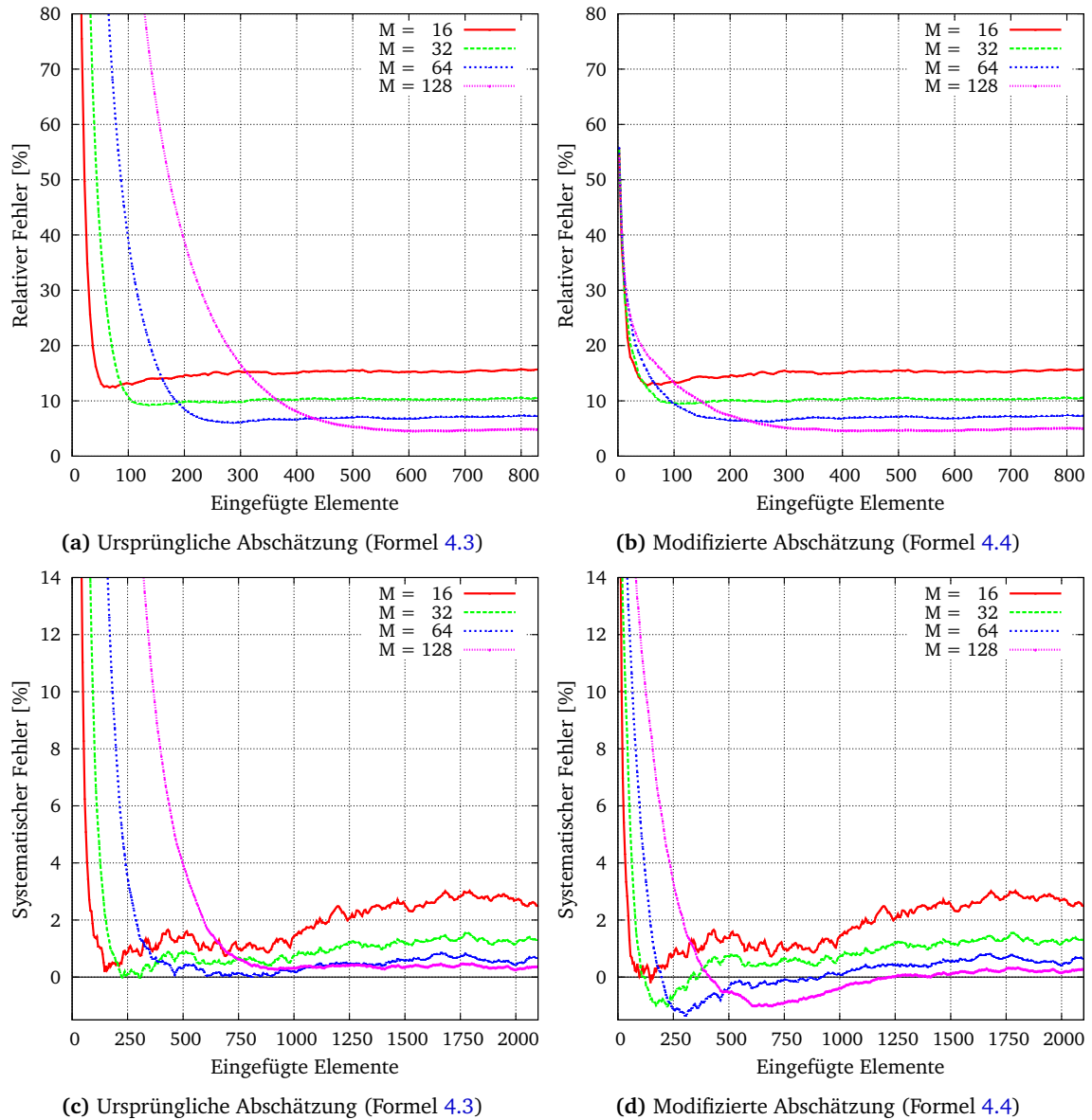


**Abbildung 4.3:** Der relative Fehler und die Standardabweichung von PCSA in Abhängigkeit von der Anzahl der verwendeten FM-Sketches.

etwa 5,3 %. Die Anzahl der FM-Sketches beeinflusst somit den Wertebereich ebenso wie die Genauigkeit. Dementsprechend eignet sich PCSA vorwiegend zur Abschätzung der Kardinalität von sehr großen Mengen.

Laut [55] erreicht die PCSA-Abschätzung erst ab etwa  $15 \cdot M$  Elementen den angegebenen mittleren Schätzfehler. Um eine Verzerrung aufgrund zu weniger Elemente zu vermeiden, werden in der zuvor präsentierten Evaluation nur die Ergebnisse ab  $40 \cdot M$  Elementen berücksichtigt. Für einen realen Einsatz ist allerdings das Verhalten der Abschätzung bei wenig gefüllten Sketches von großer Bedeutung, sodass dieser Wertebereich im Folgenden separat untersucht wird. Dafür wird der Wertebereich in Intervalle zu je fünf Elementen aufgeteilt. Für jedes Intervall wird der Mittelwert von  $|F_r|$  und  $F_r$  über tausend Simulationsdurchläufe gebildet. Abbildung 4.4 zeigt sowohl die Ergebnisse der ursprünglichen Abschätzung als auch die Ergebnisse der in Formel 4.4 dargestellten Modifikation.

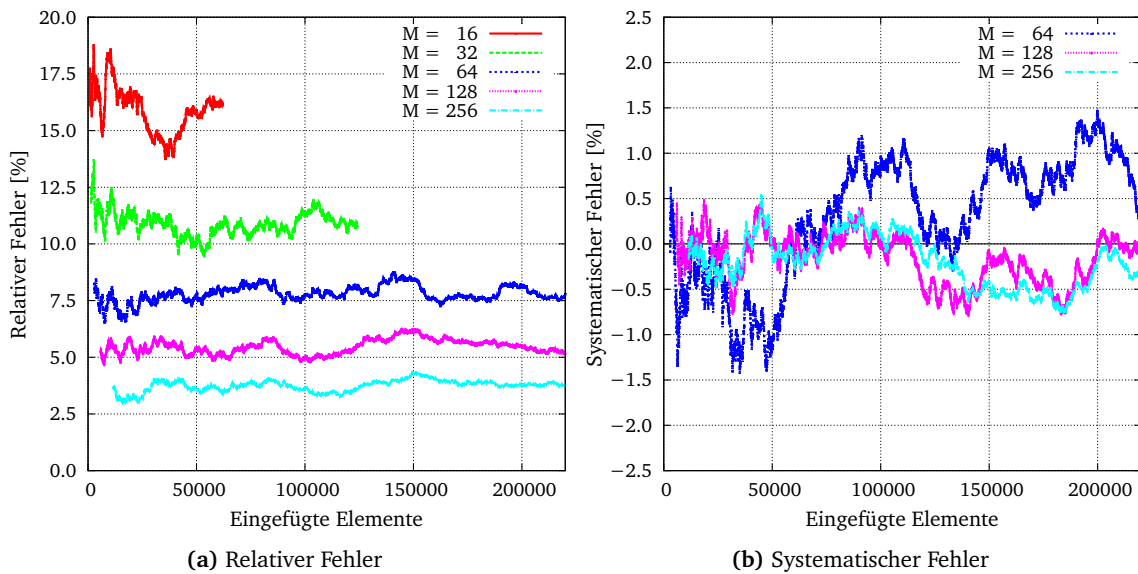
In der ersten Zeile der Abbildung ist der relative Fehler beider Abschätzungen zu sehen. Bei  $M = 16$  stabilisiert sich der relative Fehler nach etwa 45 eingefügten Elementen auf circa 15,7 %. Diese Konfiguration benötigt somit etwa drei Elemente pro Sketch, um die bestmögliche Genauigkeit zu erreichen. Allerdings erzielt die Abschätzung für die ersten 15 Elemente einen relativen Fehler von deutlich über 100 % und ist somit nahezu wertlos. Bei  $M = 64$  und  $M = 128$  werden insgesamt mindestens 240 beziehungsweise 540 Elemente benötigt, was im Durchschnitt 3,75 beziehungsweise 4,22 Elementen pro FM-Sketch entspricht. Zwar erhöht sich die durchschnittliche Mindestanzahl der Elemente pro Sketch mit steigender Sketchanzahl, aber alle ermittelten Schwellenwerte liegen deutlich unter den angegebenen  $15 \cdot M$ . Die in Formel 4.4 dargestellte Modifikation reduziert die Mindestanzahl an eingefügten Elementen



**Abbildung 4.4:** Der relative und der systematische Fehler der ursprünglichen sowie der modifizierten Abschätzung in Abhängigkeit von der Anzahl der bereits eingefügten Elemente.

signifikant. So werden beispielsweise bei  $M = 64$  und  $M = 128$  nur noch 190 beziehungsweise 350 Elemente benötigt, um den relativen Fehler zu stabilisieren. Für viele Anwendungen wird allerdings der Umstand bedeutender sein, dass der relative Fehler unterhalb der Mindestanzahl im Durchschnitt deutlich niedriger ist als bei der ursprünglichen Abschätzung.

In der zweiten Zeile von Abbildung 4.4 ist der systematische Fehler über einen im Vergleich zur ersten Zeile vergrößerten Wertebereich zu sehen. Wie bereits beim relativen Fehler kann die modifizierte Abschätzung auch den systematischen Fehler in den ersten Intervallen reduzieren. Dabei kommt es im weiteren Verlauf zu einer zeitweiligen Unterschätzung. Bei  $M = 64$  und  $M = 128$  erscheint diese Unterschätzung im Vergleich zu den Schwankungen innerhalb des dargestellten Wertebereichs relativ stark ausgeprägt zu sein. In Abbildung 4.5 sind beide Fehler über einen wesentlich größeren Wertebereich hinweg eingezeichnet. Dadurch wird deutlich, dass der systematische Fehler auch bei einer hohen Anzahl von FM-Sketches um bis zu einem Prozentpunkt schwankt. Deshalb ist die zeitweise Unterschätzung der modifizierten Formel nahezu bedeutungslos.



**Abbildung 4.5:** Der relative und der systematische Fehler von PCSA über den gesamten Wertebereich.

Wie in Abbildung 4.5 zu sehen ist, bleibt der relative Fehler trotz leichter Schwankungen über einen großen Wertebereich hinweg weitgehend stabil und ist somit nicht von der Anzahl der bisher eingefügten Elemente abhängig. Der systematische Fehler wiederum unterliegt deutlich höheren Schwankungen, die sich allerdings nahezu ausgleichen. Beispielsweise beträgt der durchschnittliche systematische Fehler bei  $M = 64$  nur 0,36 % und bei  $M = 128$  nur 0,09 %.

### 4.1.3 Weiterentwicklungen

PCSA benötigt  $\mathcal{O}(\log n)$  Bit Speicherplatz, um bis zu einer Kardinalität von  $n$  zu zählen. *Log-Log* [47] und *HyperLogLog* [54] sind Weiterentwicklungen von PCSA, die den benötigten Speicherplatz auf  $\mathcal{O}(\log \log n)$  reduzieren und sich nur in der Berechnung der Kardinalitätsabschätzung voneinander unterscheiden. Die verwendete Datenstruktur besteht aus  $M$  Sketches mit jeweils  $L$  Bits, wobei im Unterschied zu PCSA die Sketches als Zähler interpretiert werden. Des Weiteren werden eine gleich verteilte Hashfunktion  $hash(x)$  und die bereits vorgestellte Hilfsfunktion  $l_1(x)$  benötigt. Anstatt einen Sketch anhand des ganzzahligen Rests einer Division auszuwählen, werden in den Weiterentwicklungen die ersten Bits des Hashwerts direkt als Index für das Array mit den Sketches interpretiert. Der im ausgewählten Sketch hinterlegte Zähler wird mit  $j$  bezeichnet. In den verbleibenden Bits des Hashwerts wird mit Hilfe von  $l_1(x)$  die Position der ersten Eins von links bestimmt, welche im Folgenden mit  $i$  bezeichnet wird. Ist  $i + 1$  größer als  $j$ , wird der Inhalt des aktuellen Sketches auf  $i + 1$  erhöht. Beträgt der Hashwert des einzufügenden Elements beispielsweise 01001011, wird bei  $M = 4$  anhand der ersten beiden Bits der zweite Sketch ausgewählt. Für die verbleibenden Bits gilt  $l_1(001011) = 2$ . Ist der im zweiten Sketch gespeicherte Zähler kleiner als  $2 + 1$ , wird der Inhalt entsprechend aktualisiert.

Im Unterschied zu den Weiterentwicklungen wird bei FM-Sketches für jede erstmalig auftretende Position das entsprechende Bit auf eins gesetzt. Wie bereits in Formel 4.1 dargestellt, beträgt die Wahrscheinlichkeit, dass im Hashwert die erste Eins von links an Position  $i$  auftritt  $2^{-(i+1)}$ . Nachdem einige Elemente in einen FM-Sketch eingefügt wurden, ist an dessen Anfang mit hoher Wahrscheinlichkeit eine Folge von Einsen zu finden. Anhand der Länge dieser Folge kann gemäß Formel 4.2 die Anzahl der eingefügten Elemente abgeschätzt werden. Wenn der Sketch beispielsweise aus den Bits  $S = 11010000$  besteht, bedeutet dies, dass bisher nur Elemente eingefügt worden sind, für deren Hashwerte die Funktion  $l_1(x)$  null, eins oder drei zurückgegeben hat. Da bei der Abschätzung von PCSA ohnehin nur die Länge der ununterbrochenen Folge von Einsen am Anfang eines FM-Sketches verwendet wird, speichern die Weiterentwicklungen nur noch die größte bisher aufgetretene Position als Zahl. In dem konkreten Beispiel bedeutet dies, dass anstatt des Sketches nur noch die Zahl  $4 + 1$  gespeichert wird. Die Inkrementierung wird dabei nur verwendet, um die hinterlegte Zahl von einem leeren Sketch zu unterscheiden. Auf diese Weise wird gewissermaßen in jedem Sketch nur noch die niedrigste Eintrittswahrscheinlichkeit aller bisher eingefügter Elemente gespeichert.

Im Vergleich zu PCSA werden bei den Weiterentwicklungen weniger Informationen gespeichert, sodass sich die Genauigkeit der Kardinalitätsabschätzung bei LogLog auf  $1,30/\sqrt{M}$  und bei HyperLogLog auf  $1,04/\sqrt{M}$  verschlechtert. Allerdings können  $n$  unterschiedliche Elemente anstatt in  $\mathcal{O}(\log n)$  Bit mit den Weiterentwicklungen in nur noch  $\mathcal{O}(\log \log n)$  Bit gezählt werden. Dementsprechend kann die niedrigere Genauigkeit durch eine Erhöhung der Anzahl der Sketches bei gleich bleibendem Speicherplatz mehr als ausgeglichen werden. Beispielsweise beträgt bei  $M = 128$  der mathematisch hergeleitete mittlere Schätzfehler von PCSA 6,9 %. Wenn pro Sketch 15 Bits verwendet werden, benötigt die gesamte Datenstruktur 240 Byte.

Bei dieser Konfiguration kann  $l_1(x)$  während des Einfügeprozesses Werte zwischen 0 und 14 annehmen. Inklusive einer zusätzlichen Zahl zur Repräsentation eines leeren Sketches ist dieser Wertebereich mit 4 Bit durchzählbar. Bei gleich bleibenden Speicherplatz können deshalb anstatt der 128 FM-Sketches 480 Zähler verwendet werden, die mit Hilfe der HyperLogLog-Abschätzung einen mittleren Schätzfehler von 4,7 % ermöglichen.

Obwohl die niedrigere Genauigkeit durch ein Erhöhen der Anzahl der Sketches mehr als ausgeglichen werden kann, werden im Folgenden ausschließlich FM-Sketches verwendet. Die Ursache hierfür liegt darin, dass die in Abschnitt 4.2 erläuterte und analysierte Methode zur verteilten Datenaggregation das bitweise Speichern der Rückgabewerte von  $l_1(x)$  benötigt. Aufgrund der limitierten Bandbreite in einem drahtlosen Ad-hoc-Netzwerk ist auch die zu übertragende Datenmenge von hoher Bedeutung. Aus diesem Grund wird in [134] unter anderem die Entropie von FM-Sketches und HyperLogLog-Sketches untersucht. Die Ergebnisse zeigen, dass bei einer Konfiguration mit vergleichbarer Genauigkeit auch die Entropie beider Datenstrukturen in etwa auf dem gleichen Niveau liegt. Da bei einer optimalen Kompression (siehe Abschnitt 4.2.2) beide Datenstrukturen in etwa den gleichen Speicherplatz benötigen, führt die Fokussierung auf FM-Sketches nicht zwangsläufig zu einer signifikanten Erhöhung des zu erwartenden Datenvolumens.

### 4.1.4 Verbesserung der Schätzung

Die im vorherigen Abschnitt erläuterten Weiterentwicklungen haben vorrangig das Ziel, bei gleich bleibendem Speicherbedarf mehr Elemente zählen zu können. Im Unterschied dazu wird in diesem Abschnitt untersucht, inwieweit die Genauigkeit der Abschätzung verbessert werden kann. Der Aufbau der Datenstruktur und der Einfügealgorithmus werden dementsprechend nicht geändert, sodass die verbesserte Abschätzung auch für die im weiteren Verlauf der Arbeit beschriebene dezentrale Aggregation eingesetzt werden kann.

Ausgehend von einem einzelnen FM-Sketch kann die Anzahl der bisher eingefügten Elemente, wie weiter oben erwähnt, mit  $C(S) = 2^{l_0(S)}/p$  (Formel 4.2) abgeschätzt werden. Gemäß dieser Formel werden alle Informationen, die nach der ersten Null von links gespeichert sind, nicht direkt für die Abschätzung genutzt. Bei dem Sketch  $S = 11101100$  mit  $l_0(S) = 3$  wird beispielsweise das Bitmuster ab der dritten Position im Rahmen der Schätzung nicht berücksichtigt. Im Durchschnitt wird dieser Sachverhalt zwar teilweise durch den Korrekturfaktor  $p = 0,775351$  ausgeglichen, dennoch kann die direkte Nutzung dieser Zusatzinformationen die Genauigkeit der Abschätzung verbessern.

Prinzipiell kann – abhängig von der Anzahl der eingefügten Elemente – berechnet werden, mit welcher Wahrscheinlichkeit ein bestimmtes Bitmuster  $S$  auftritt. Aufgrund der zahlreichen bedingten Wahrscheinlichkeiten ist die Formel allerdings komplex und müsste für jedes mögliche Bitmuster neu hergeleitet werden. Da der Schwerpunkt dieser Arbeit nicht im mathematischen Bereich liegt, der jeweilige Kurvenverlauf der Wahrscheinlichkeit sich aber zur Wahl eines geeigneten Schätzers anbietet, werden die benötigten Daten empirisch gewonnen. Ein Sketch der Länge  $L$  kann  $2^L$  unterschiedliche Zustände annehmen. Gleichzeitig steigt mit der

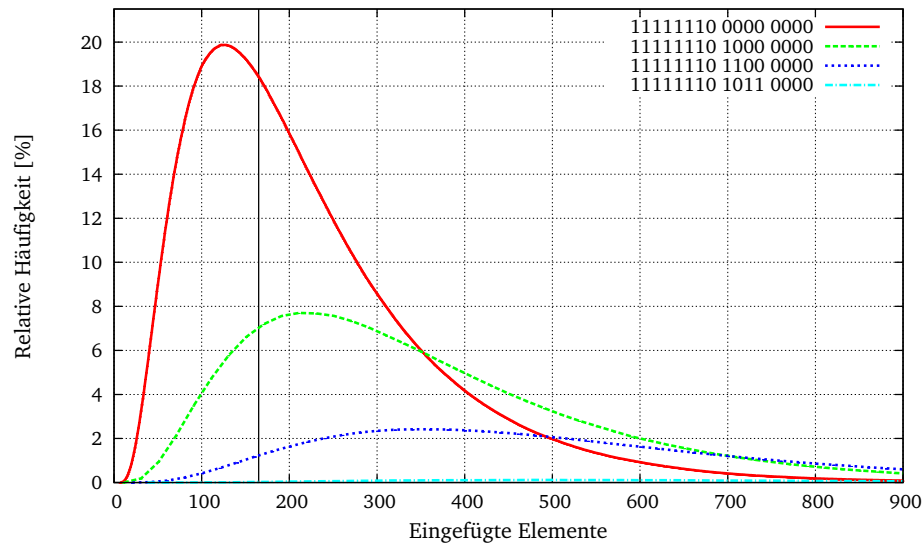
Länge auch die Anzahl der Elemente, die gezählt werden können. Dementsprechend vervielfachen sich mit steigender Länge der Sketches sowohl die Anzahl der möglichen Bitmuster als auch der Rechenaufwand, der für die empirische Bestimmung der Wahrscheinlichkeiten benötigt wird. Die in den vorherigen Abschnitten verwendete Konfiguration  $L = 16$  ist für viele Einsatzszenarien mehr als ausreichend und kann gleichzeitig noch mit vertretbarem Aufwand empirisch evaluiert werden.

Zu Beginn eines Simulationsdurchlaufs erzeugt das entwickelte Programm einen leeren FM-Sketch, dem sukzessive so lange Elemente hinzugefügt werden, bis alle Bits auf eins gesetzt sind. Als einzufügende Elemente werden zufällige Zahlen zwischen 0 und 65.535 verwendet, die mit Hilfe des Pseudozufallszahlengenerators *Mersenne-Twister* [109] generiert werden. Die Anzahl der bereits eingefügten Elemente wird im Folgenden mit  $x$  bezeichnet. Immer wenn durch den aktuellen Einfügevorgang ein bisher ungesetztes Bit gesetzt wird, speichert das Programm für das neue Bitmuster die Zahl  $x$  als Untergrenze ab. Diese gibt an, bei wie vielen eingefügten Elementen das Muster während dieses Simulationsdurchgangs zum ersten Mal aufgetreten ist. Analog zu diesem Vorgang wird für das bisherige Bitmuster die Obergrenze von  $x - 1$  gespeichert. Nach einer Million Simulationsdurchläufen werden für jedes Bitmuster die kleinste Untergrenze und die größte Obergrenze gesucht. Diese beiden Grenzen spezifizieren den Wertebereich, in dem das jeweilige Bitmuster aufgetreten ist. Durch das Überprüfen der gespeicherten Unter- und Obergrenzen kann nun für jede Zahl  $x$  innerhalb dieses Wertebereichs ermittelt werden, wie oft das jeweilige Bitmuster nach exakt  $x$  Einfügevorgängen aufgetreten ist.

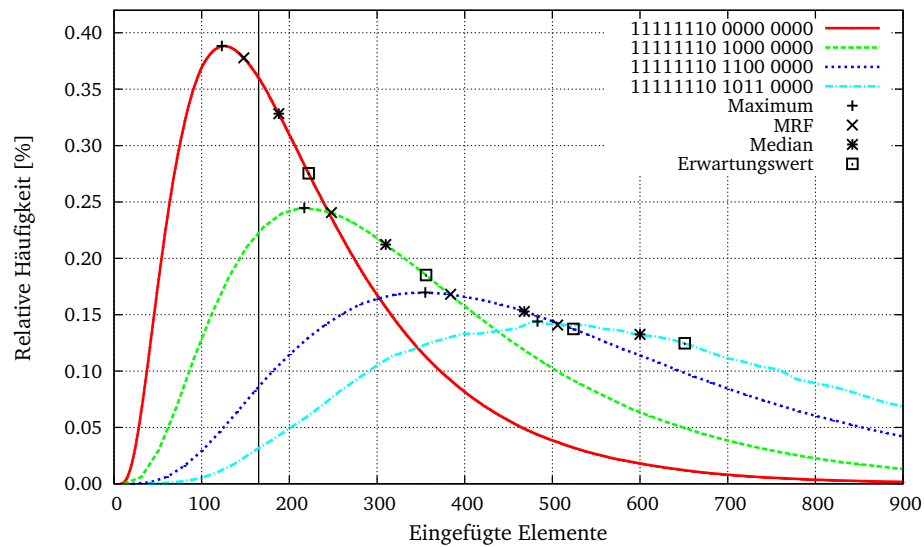
Abbildung 4.6 zeigt die relative Häufigkeit von vier Bitmustern in Abhängigkeit von der Anzahl der bereits eingefügten Elemente. Üblicherweise werden diskrete Ereignisse als Punktbeziehungweise Balkendiagramm dargestellt. In den präsentierten Abbildungen ist allerdings die Datendichte auf der x-Achse ohnehin so hoch, dass die Punktemenge als kontinuierliche Linie wahrgenommen werden würde. Daher wurden die Einzeldaten direkt als Linie gezeichnet, sodass eine bessere Unterscheidung der einzelnen Verläufe gewährleistet ist. Der verwendete Stichprobenumfang ist für die meisten Bitmuster ausreichend groß, um sicherzustellen, dass die relativen Häufigkeiten gemäß dem Gesetz der großen Zahlen nahezu den realen Wahrscheinlichkeiten entsprechen. Aus diesem Grund wird im Folgenden die relative Häufigkeiten eines Ereignisses mit seiner Eintrittswahrscheinlichkeit gleichgesetzt.

In der oberen Teilabbildung entspricht die relative Häufigkeit dem Verhältnis zwischen der absoluten Häufigkeit und der Anzahl der Simulationsdurchläufe. Dementsprechend gibt der y-Wert an, mit welcher Wahrscheinlichkeit nach exakt  $x$  Einfügevorgängen das jeweilige Bitmuster im FM-Sketch vorgefunden werden kann. Gemäß der Abschätzung 4.2 wurden in den FM-Sketch  $S = 1111111000000000$  mit  $l_0(S) = 7$  circa 165 Elemente eingefügt. Da die Abschätzung nur die Länge der Folge von Einsen am Anfang des FM-Sketches berücksichtigt, ergibt sie für die anderen in der Abbildung dargestellten Bitmuster exakt den gleichen Wert. Die maximale Eintrittswahrscheinlichkeit der anderen Bitmuster ist im Vergleich zu  $S$  niedriger. Des Weiteren verteilen sich auch die Eintrittsmöglichkeiten auf einen größeren Wertebereich.





(a) Absolute Häufigkeit in Relation zur Anzahl der Simulationsdurchläufe



(b) Absolute Häufigkeit in Relation zur Summe aller absoluten Häufigkeiten des jeweiligen Bitmusters

**Abbildung 4.6:** Relative Häufigkeit von vier Bitmustern innerhalb eines FM-Sketches in Abhängigkeit von der Anzahl der bereits eingefügten Elemente.



Dennoch ist aus dem Verlauf der anderen Bitmuster ersichtlich, dass diese mit einer hohen Wahrscheinlichkeit erst bei einem höheren  $x$ -Wert auftreten. Aus diesem Grund kann die Abschätzung verbessert werden, indem für jedes Bitmuster ein separater Schätzwert bestimmt wird.

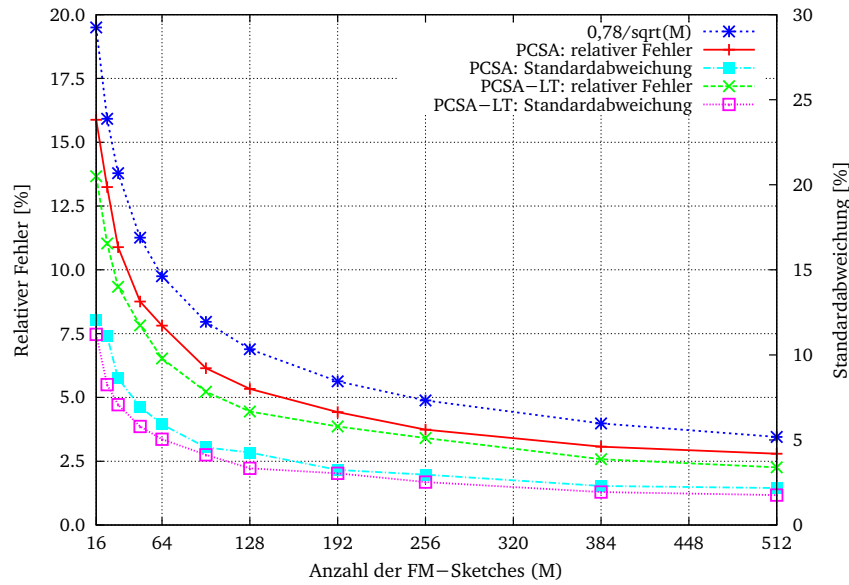
Die auf der  $y$ -Achse der unteren Teilabbildung dargestellte relative Häufigkeit entspricht dem Verhältnis zwischen der absoluten Häufigkeit nach  $x$  Einfügevorgängen und der Summe aller absoluten Häufigkeiten des jeweiligen Bitmusters. Dementsprechend ist die Summe aller relativen Häufigkeiten eines Bitmusters gleich eins beziehungsweise 100 %. Für jedes Bitmuster wurden unter anderem das Maximum, der Erwartungswert und der Median der Verteilung eingezeichnet. Der Erwartungswert entspricht der Summe der Produkte aus den relativen Häufigkeiten ( $y$ -Achse) und der jeweiligen Anzahl der eingefügten Elemente ( $x$ -Achse). Gemäß dieser Definition konvergiert der durchschnittliche Fehler  $F$  zwischen der realen Anzahl von eingefügten Elementen und dem als Schätzer verwendeten Erwartungswert gegen null, wodurch der systematische Fehler  $|F|$  der Schätzung minimiert wird. Da allerdings nicht der absolute, sondern der relative Fehler (Formel 4.5) für die vorliegende Datenstruktur von Bedeutung ist, wird für jedes Bitmuster derjenige Wert als Schätzer verwendet, der im Durchschnitt den kleinsten relativen Fehler  $|F_r|$  verursacht. In Abbildung 4.6 wurden diese Schätzwerte mit dem Kürzel MRF (minimaler relativer Fehler) eingezeichnet.

Die Schätzwerte aller Bitmuster werden in der Lookup-Tabelle  $LT(S)$  bereitgehalten. Da von den theoretisch möglichen 65536 Bitmustern während der Evaluation nur 3311 aufgetreten sind, wird für die verbleibenden 62225 die bisherige Funktion zur Abschätzung verwendet. Diese Bitmuster – beispielsweise 01111111111111 – sind äußerst unwahrscheinlich und haben dementsprechend nur einen geringen Einfluss auf die durchschnittliche Genauigkeit der Abschätzung. Die vergleichsweise niedrige Anzahl der aufgetretenen Bitmuster ist allerdings ein Hinweis darauf, dass der benötigte Speicherplatz im Durchschnitt deutlich über dem eigentlichen Informationsgehalt der Datenstruktur liegt.

Die Anzahl der in einem FM-Sketch eingefügten Elemente kann mit der Lookup-Tabelle signifikant besser abgeschätzt werden als mit der Funktion  $C(S)$ . Um allerdings zufriedenstellende Ergebnisse zu erzielen, müssen wie bei PCSA mehrere FM-Sketches miteinander kombiniert werden. Dabei liegt es nahe, für jeden FM-Sketch mit Hilfe der Lookup-Tabelle den entsprechenden Schätzer zu bestimmen und dann die Summe zu bilden. Der durchschnittliche relative Fehler dieser Strategie ist vertretbar, liegt aber klar über den Ergebnissen von PCSA. Des Weiteren ist insbesondere die Standardabweichung des relativen Fehlers extrem hoch. Aus diesem Grund werden die Teilergebnisse wie bei PCSA logarithmisch gemittelt. Hierfür wird folgende modifizierte Abschätzung verwendet:

$$C(S_0, \dots, S_{M-1}) = \frac{M}{p_{LT}} \cdot 2^{\sum_{j=0}^{M-1} (\log_2 \cdot LT(S_j)) / M}. \quad (4.6)$$

Um den benötigten Korrekturfaktors  $p_{LT}$  zu bestimmen, berechnet das Evaluationsprogramm den Mittelwert derjenigen Korrekturfaktoren, die für die Einzelergebnisse gemäß Formel 4.6 den kleinsten Fehler verursachen. Hierbei wurde für jede verwendete Konfiguration



**Abbildung 4.7:** Relativer Fehler und Standardabweichung von PCSA-LT in Abhängigkeit von der Anzahl der verwendeten FM-Sketches.

von  $M$  ein separater Korrekturfaktor bestimmt. Die Werte sind aber bei allen Konfigurationen nahezu identisch und bilden den Mittelwert  $p_{LT} = 1,2366$ . Im Folgenden wird die Version mit der Lookup-Tabelle und der angepassten Schätzung PCSA-LT genannt.

Da sich die Eingabedaten während der Berechnung des Korrekturfaktors und der Bestimmung des mittleren relativen Fehlers unterscheiden sollten, werden in beiden Vorgängen unterschiedliche Pseudozufallszahlengeneratoren und verschiedene Strategien zur Generierung der Zeichenketten verwendet. Um sicherzustellen, dass diese Vorgehensweise nicht dennoch zu einem Vorteil von PCSA-LT (empirisch ermittelte Korrekturfaktoren in Abhängigkeit von der Anzahl der Sketches) gegenüber PCSA (mathematisch hergeleiteter Korrekturfaktor für alle Konfigurationen) führt, wurden auch für PCSA optimierte Korrekturfaktoren evaluiert. Die mit diesen Korrekturfaktoren erzielten Ergebnisse unterscheiden sich allerdings nicht signifikant von den in Abschnitt 4.1.2 präsentierten Resultaten, sodass im Folgenden ausschließlich die Letzteren zu Vergleichszwecken herangezogen werden.

Abbildung 4.7 zeigt den relativen Fehler und die Standardabweichung von PCSA-LT in Abhängigkeit von der Anzahl der verwendeten FM-Sketches. Die Simulationsergebnisse belegen, dass die modifizierte Abschätzung die Genauigkeit gegenüber dem Basisverfahren verbessert. So erreicht beispielsweise PCSA-LT mit  $M = 64$  eine Genauigkeit, die PCSA erst bei  $M = 96$  erzielt. In diesem Beispiel können bei gleich bleibender Genauigkeit etwa 33 % des Speicherplatzes eingespart werden. Allerdings verringert sich der Abstand zwischen beiden Verfahren mit steigender Sketchzahl. Die Ursache dafür liegt in dem Umstand, dass sich die einzelnen Schätzfehler der FM-Sketches zunehmend im Mittel ausgleichen. Bei dezentralen Anwendungen ist allerdings das benötigte Übertragungsvolumen von großer Bedeutung, sodass tenden-

ziell eher weniger FM-Sketches parallel eingesetzt werden. Deshalb ist die in diesem Abschnitt vorgestellte Strategie insbesondere für derartige Szenarien geeignet.

Wie in der vorherigen Evaluation teilt auch hier das Programm den gesamten Wertebereich in Intervalle auf und bestimmt für jedes von ihnen die relevanten Kenngrößen. Da aber der Kurvenverlauf beider Abschätzungen sehr ähnlich und nur um wenige Prozentpunkte verschoben ist, wird an dieser Stelle auf eine Darstellung verzichtet. Der systematische Fehler gleicht sich wie bei PCSA über den Wertebereich nahezu aus und beträgt beispielsweise bei  $M = 64$  nur 0,38 % und bei  $M = 128$  nur 0,03 %.

Die vorliegende Implementierung verwendet eine Lookup-Tabelle mit den 3311 zuvor evaluierten Schätzwerten. Für die verbleibenden Bitmuster sind in der Tabelle die Werte gemäß Formel 4.3 hinterlegt. Da jeder Schätzwert 16 Bit benötigt, beträgt der Speicherbedarf der Tabelle  $2^{16} \cdot 2 \text{ Byte} = 128 \text{ Kilobyte}$ . Werden die 3311 Schätzwerte hingegen in einem assoziativen Datenfeld gespeichert und die anderen Schätzwerte nur im Bedarfsfall berechnet, so sinkt der Speicherbedarf auf wenige Kilobyte. Bei heute gängigen Computersystemen ist dieser Speicherbedarf unerheblich, in Sensornetzen oder eingebetteten Systemen hingegen möglicherweise ein Ausschlusskriterium. Für eine solche Einsatzumgebung wäre es nützlich, wenn die Schätzwerte anhand einer Formel berechnet werden könnten. Zu diesem Zweck könnten die Schätzwerte mit Hilfe der Wahrscheinlichkeitsrechnung auf mathematischem Weg hergeleitet werden. Alternativ ist es auch möglich, aus den empirisch bestimmten Werten eine hinreichend genaue Berechnungsfunktion herzuleiten.

**Tabelle 4.1:** Exemplarische Einträge aus der von PCSA-LT verwendeten Lookup-Tabelle.

| Bitmuster               | Schätzwert | Bitmuster               | Schätzwert | Bitmuster               | Schätzwert |
|-------------------------|------------|-------------------------|------------|-------------------------|------------|
| 111111 0000000000       | 74         | 111111 0100000000       | 124        | 111111 0101000000       | 183        |
| 11111111 0000000000     | 148        | 11111111 0100000000     | 248        | 11111111 0101000000     | 369        |
| 1111111111 0000000000   | 295        | 1111111111 0100000000   | 497        | 1111111111 0101000000   | 740        |
| 11111111111 0000000000  | 589        | 11111111111 0100000000  | 991        | 11111111111 0101000000  | 1477       |
| 111111111111 0000000000 | 1176       | 111111111111 0100000000 | 1986       | 111111111111 0101000000 | 2970       |

In Tabelle 4.1 sind exemplarisch Schätzwerte für ähnliche Bitmuster aufgelistet. Aus Performancegründen werden beim Bestimmen der Schätzwerte nur ganzzahlige Werte berücksichtigt, sodass die angegebenen Ergebnisse nicht exakt dem jeweiligen Optimum entsprechen. Dennoch lässt sich deutlich erkennen, dass sich die Schätzwerte bei jeder Verschiebung nahezu exakt verdoppeln. Durch die Wahl einer geeigneten Modellfunktion und eine geschickte Anordnung der Schätzwerte können möglicherweise sogar alle Werte durch eine Funktion approximiert werden. Wenn die Rückgabewerte der approximierenden Funktion den gesuchten Schätzwerten entsprechen, ist mit der Anordnung der Schätzwerte Letzteren das Schema zur Erzeugung der zugehörigen Parameter gemeint. Immer wenn der Schätzwert eines FM-Sketches gesucht wird, muss aus dem jeweiligen Bitmuster zumindest ein Wert generiert werden, welcher der Schätzfunktion als Parameter übergeben werden kann. Das Bitmuster direkt als Zahl zu interpretieren ist für diesen Zweck ungeeignet, da diese Zahlen im Zusammenspiel mit den zugehörigen Schätzwerten keinen stetigen Kurvenverlauf erzeugen. Möglicherweise

ist es sinnvoll, zwei Parameter zu verwenden. Der erste könnte beispielsweise die Länge der Folge von Einsen am Anfang des FM-Sketches sein. Die nach dieser Folge kommenden vier Bits repräsentieren das jeweilige Muster und könnten der Schätzfunktion als zweiter Parameter übergeben werden. Für die eigentliche Optimierung kann beispielsweise der *Levenberg-Marquardt-Algorithmus* [100] herangezogen werden, der ausgehend von einer Modellfunktion und einer Menge von Tupeln (der aus den Bitmustern resultierende Parameter sowie der zugehörigen Schätzwerte) näherungsweise die optimalen Parameter innerhalb der Modellfunktion bestimmt.

### 4.2 Kooperative Aggregation von orts- und zeitbezogenen Daten

Die in Abschnitt 2.2 exemplarisch vorgestellten kooperativen Anwendungen haben die Gemeinsamkeit, dass sie dezentral vorliegende Informationen wie beispielsweise Sensordaten oder Beobachtungen austauschen und miteinander verrechnen. Im Bereich der Sensornetze werden typischerweise von einem mit der Außenwelt verbundenen Rechner, der sogenannten Datensinke, Anfragen an die Knoten im Beobachtungsgebiet gestellt. Nachdem eine Anfrage an alle relevanten Knoten weitergeleitet wurde, schicken diese die gewünschten Informationen zurück an die Datensinke. Um die Anzahl der zu übertragenden Pakete zu minimieren und somit auch den meist limitierten Energievorrat zu schonen, senden die Knoten in der Regel die Informationen nicht per Ende-zu-Ende-Kommunikation an die Datensinke, sondern verwenden einen bei der Verbreitung der Anfrage aufgebauten Spannbaum. Wenn beispielsweise die durchschnittliche Temperatur im Beobachtungsgebiet ermittelt werden soll, so übertragen die Blätter des Spannbaums die gemessenen Temperaturen an ihre Väter. Die Väter addieren die Temperatur von all ihren Kindern und übertragen diese Summe sowie die Anzahl der Temperaturmessungen wiederum an ihre Väter. Dieser Schritt wird so lange wiederholt, bis die Informationen an der Wurzel, also der Datensinke, angekommen sind. Dort kann anhand der beiden Summen die durchschnittliche Temperatur errechnet werden.

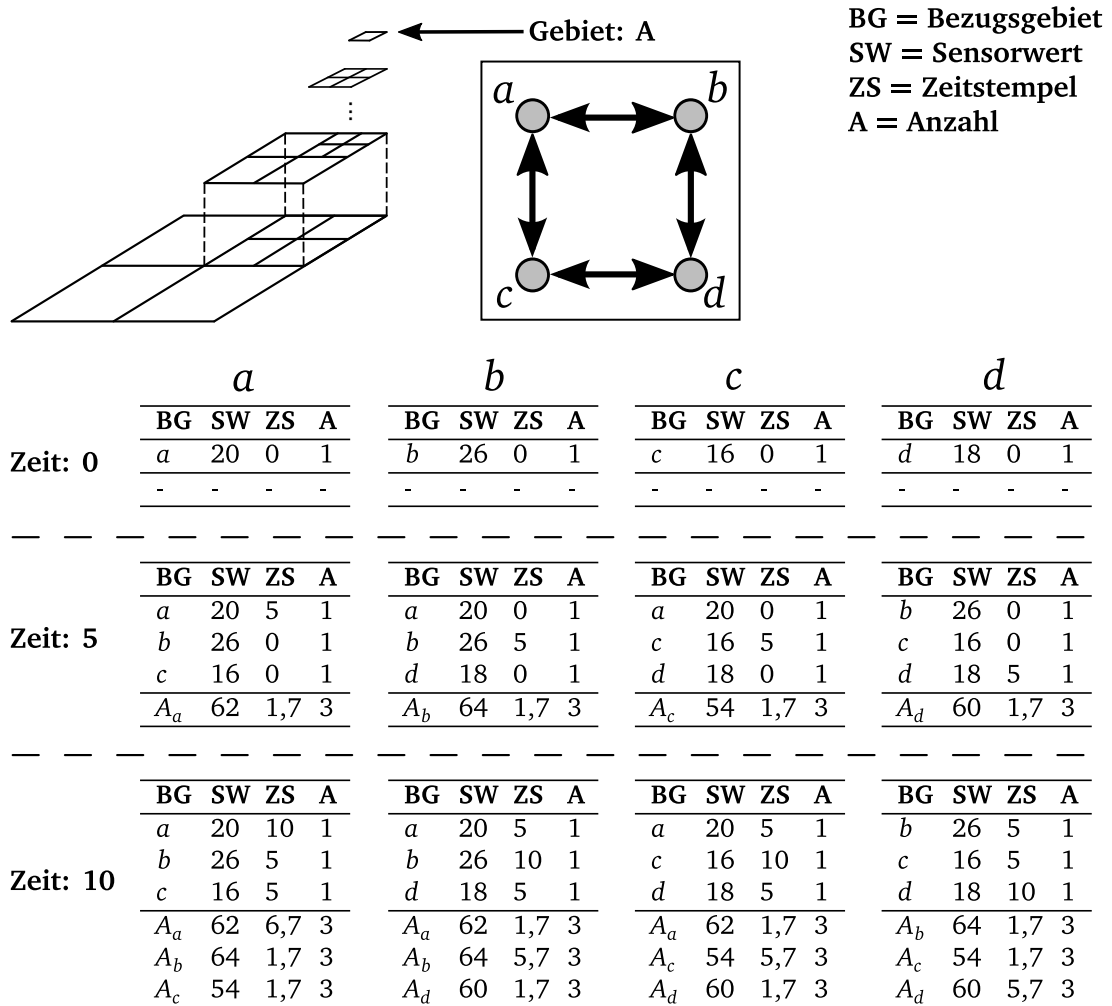
Sensorknoten besitzen meist nur einen begrenzten Energievorrat, sodass ihre Lebensdauer primär von der Anzahl der übertragenen Pakete abhängt. Da der Kommunikationsaufwand aber nicht auf alle Knoten gleich verteilt anfällt, ist auch die Lebensdauer der Knoten uneinheitlich. Dementsprechend wird ein einmal aufgebauter Spannbaum nach einer gewissen Zeit ungültig, sodass nicht mehr alle Informationen korrekt zur Datensinke weitergeleitet werden. Um diesem Problem entgegenzuwirken, kann von vorneherein eine Mehrwegeweiterleitung verwendet werden. Dieser Vorgang ist allerdings aufwendig, da bei der dezentralen Summierung der Einzelwerte darauf geachtet werden muss, dass jeder Wert nur einmal berücksichtigt wird. Um Aggregierungsfehler zu vermeiden, müssten dementsprechend die prinzipiell unabhängigen Weiterleitungsprozesse aufwendig miteinander koordiniert werden. Aus diesem Grund wird in [38] vorgeschlagen, für den Datenaustausch und die dezentrale Bestimmung von einfachen Aggregierungsfunktionen wie SUM oder COUNT FM-Sketches zu verwenden.

Anstatt die Summe der Einzelwerte als Zahl zur Datensenke weiterzuleiten, werden in diesem Ansatz FM-Sketches ausgetauscht. Der Knoten  $y$  kann die Zahl  $x$  in einem FM-Sketch für eine Summenberechnung hinzufügen, indem er beispielsweise  $x$  unterschiedliche Elemente als Zeichenketten der Form " $y : 1$ ",  $\dots$ , " $y : x$ " einfügt. Wie in den vorherigen Abschnitten erläutert, können mehrere FM-Sketches durch eine bitweise OR-Operation miteinander verknüpft werden, wobei der resultierende FM-Sketch die Vereinigungsmenge aller eingefügten Elemente enthält. Dementsprechend muss bei einer Mehrwegeweiterleitung nicht darauf geachtet werden, an welcher Stelle im Netzwerk die Beobachtungen miteinander verrechnet werden und ob dabei möglicherweise Einzelwerte mehrfach berücksichtigt werden. Nachdem die FM-Sketches zur Datensenke weitergeleitet wurden, kann diese mit Hilfe der Schätzfunktion die Anzahl der eingefügten Elemente näherungsweise bestimmen und so die Summe aller Einzelwerte abschätzen.

Im vorherigen Beispiel aus dem Bereich der Sensornetze werden die ortsbezogenen Sensordaten zu einem bestimmten Zeitpunkt oder in einem regelmäßigen Intervall von der Datensenke abgefragt. Die zur Datensenke weitergeleiteten Sensorwerte entsprechen dem jeweils aktuellen Kenntnisstand der Knoten, sodass während der Weiterleitung immer nur Sensorwerte des etwa gleichen Alters und desselben Bezugsgebiets miteinander kombiniert werden. Im Unterschied dazu verarbeiten die in Abschnitt 2.2.2 vorgestellten Anwendungen zur dezentralen Verkehrserfassung und Parkplatzsuche Informationen mit unterschiedlichem Alter und verschiedenen Bezugsgebieten. Zur Identifikation der Bezugsgebiete werden in SOTIS [162] die Straßen in eindeutig nummerierte Abschnitte unterteilt. Beim Datenaustausch werden dann nur diejenigen Basisinformationen miteinander kombiniert, die sich auf den jeweils gleichen Straßenabschnitt beziehen. Diese Strategie wird in [30] zu einem mehrstufigen Aggregierungsschema weiterentwickelt. Dabei wird das gesamte Beobachtungsgebiet als höchste Aggregierungsebene aufgefasst, die in vier gleich große, sich nicht überschneidende Untergebiete aufgeteilt wird. Diese wiederum werden nach den gleichen Regeln so lange in jeweils vier Untergebiete aufgeteilt, bis die gewünschte Gebietsgröße erreicht wurde.

Wie bei SOTIS übertragen auch in diesem Ansatz die Fahrzeuge regelmäßig die eigenen Basisinformationen wie beispielsweise die Anzahl der freien Parkplätze oder die eigene Geschwindigkeit an die benachbarten Fahrzeuge. Anhand der empfangenen Basisinformationen werden die Aggregate der niedrigsten Aggregierungsebene berechnet, welche primär aus der Summe aller gleichartigen Basisinformationen innerhalb des jeweiligen Gebiets, der Anzahl der beteiligten Knoten und dem durchschnittlichen Zeitstempel aller berücksichtigten Daten bestehen. Die Aggregate der höheren Ebenen werden durch eine Summierung der Aggregate der jeweils zugehörigen vier Untergebiete errechnet. Diese Aggregate werden abhängig von der Aggregierungsebene und der Position des Fahrzeugs ebenfalls in regelmäßigen Abständen mit den benachbarten Fahrzeugen ausgetauscht.

In Abbildung 4.8 ist eine exemplarische Aggregierung innerhalb der niedrigsten Aggregierungsebene dargestellt. Innerhalb des betrachteten Gebiets  $A$  befinden sich vier Knoten, die – wie in der Abbildung dargestellt – miteinander vernetzt sind. Zum Zeitpunkt 0 wird von je-



**Abbildung 4.8:** Dezentrale Aggregation innerhalb des kleinsten Aggregierungsgebiets:  
 Jeder Knoten überträgt nach jeweils fünf Zeiteinheiten seinen eigenen Sensorwert und das selbst berechnete Aggregat des Bezugsgebiets A. Ab Zeitpunkt 5 kennt jeder Knoten die Sensorwerte seiner Nachbarn und ab Zeitpunkt 10 auch die entsprechenden Aggregate, die allerdings nicht verlustfrei miteinander kombiniert werden können.

dem Knoten die eigene Basisinformation an die jeweiligen Nachbarn übertragen. Nach diesem Austausch kennt jeder Knoten drei Basisinformationen, die jeweils zu einem Aggregat miteinander verrechnet werden. Zum Zeitpunkt 5 werden von jedem Knoten sowohl die eigene aktualisierte Basisinformation als auch das berechnete Aggregat übertragen. Zum nächsten Übertragungszeitpunkt kennt somit jeder Knoten die im letzten Schritt übertragenen Basisinformationen und Aggregate. Da in diesen Aggregaten allerdings keine Angabe enthalten ist, welche Basisinformation von welchem Knoten stammt, können die Aggregate weder durch die anderen Aggregate des gleichen Beobachtungsgebiets noch durch die zusätzlichen Basisinformationen ergänzt werden. Gleichzeitig wird anhand der vorliegenden Basisinformationen das lokal berechnete Aggregat aktualisiert, welches dann einen neueren Zeitstempel aufweist als die anderen beiden Aggregate. Folglich können in diesem Beispiel die Aggregate der benachbarten Knoten nicht sinnvoll verwendet werden.

Da in SOTIS die Größe der Straßenabschnitte so gewählt wird, dass jeweils alle Knoten innerhalb eines Straßenabschnitts miteinander kommunizieren können, kann dieses Problem nicht auftreten, solange man von einer fehlerfreien Übertragung und einem hinreichend kleinen Austauschintervall ausgehen kann. Des Weiteren übertragen die Knoten in den besprochenen Verfahren anders als im Beispiel nicht nur die eigenen Sensorwerte, sondern auch eine Teilmenge der empfangenen Basisinformationen. Die Autoren des Ansatzes zur dezentralen Parkplatzsuche empfehlen, dass jede übertragene Nachricht mindestens so viele Basisinformationen enthalten sollte, wie es Parkplatzautomaten innerhalb des kleinsten Beobachtungsgebiets gibt. Dabei werden die empfangenen Basisinformationen abhängig von ihrem Alter und Entstehungsort sortiert, und diejenigen mit der höchsten Priorität werden für die Weiterleitung ausgewählt. Wenn allerdings die Datenerhebung nicht auf wenige Knoten beschränkt wird – in diesem Fall die Parkplatzautomaten –, ist die konkrete Anzahl der Datenquellen innerhalb des Gebiets nicht a priori bekannt. Des Weiteren ist es wahrscheinlich, dass in einem solchen Szenario nicht mehr alle Basisinformationen eines Gebiets in einem einzelnen Paket übertragen werden können. Dementsprechend tritt die im Beispiel dargestellte Problematik auch auf, wenn mehr als eine Basisinformation übertragen wird.

Die Autoren von [105] schlagen deshalb vor, anstatt der Summe der jeweiligen Basisinformationen FM-Sketches auszutauschen. Wie in dem Beispiel aus dem Bereich der Sensornetze generiert auch hier jeder Knoten so viele unterschiedliche Elemente, wie es dem Zahlenwert der Basisinformation entspricht. Damit diese Elemente für jeden Knoten unterschiedlich sind, kann für die Zeichenkettengenerierung der Identifikator des jeweiligen Knotens herangezogen werden. Der Aufbau der FM-Sketches erlaubt es den Empfängern, sowohl ihre eigenen Basisinformationen als auch andere Aggregate mit den empfangenen FM-Sketches zu kombinieren. Während im ersten Fall die lokal generierten Elemente den empfangenen FM-Sketches hinzugefügt werden, werden im zweiten Fall die entsprechenden FM-Sketches mittels einer OR-Operation miteinander kombiniert. Da es aufgrund dieser Eigenschaften ausreicht, allein die Aggregate auszutauschen, entfällt die separate Übertragung von ausgewählten Basisinformationen. In dem weiter oben besprochenen Beispiel ermöglicht es diese Vorgehensweise, dass



zum Zeitpunkt 10 jeder Knoten die drei vorliegenden Aggregate zusammenfassen kann und somit bei der nächsten Übertragung ein Aggregat sendet, welches alle vier Basisinformationen enthält. Zwar kann dabei die Summe der Basisinformationen nur näherungsweise bestimmt werden, dafür ermöglicht dieser Ansatz aber die verlustfreie Kombination von Aggregaten des gleichen Bezugsgebiets. Da die Aggregate der höheren Aggregierungsebenen ausschließlich auf den Aggregaten der vier zugehörigen Teilgebiete basieren, verbessert sich aufgrund der verlustfreien Kombination insbesondere ihre Genauigkeit.

### 4.2.1 Soft State Sketches

Ein dezentrales Aggregierungsschema muss nicht nur einmalig die Aggregate auf allen Ebenen erzeugen, sondern diese auch aktuell halten. Die Bewegung eines Knotens kann beispielsweise dazu führen, dass die entsprechende Basisinformation nun zu einem anderen Bezugsgebiet gehört und dementsprechend aus den Aggregaten des bisherigen Gebietes entfernt werden muss. Des Weiteren sind auch die eingespeisten Informationen keineswegs konstant, sodass auch ohne Knotenbewegung eine kontinuierliche Aktualisierung der Aggregate notwendig ist. Da aber in einem FM-Sketch mehrere Elemente auf die gleiche Bitposition abgebildet werden, kann ein einmal eingefügtes Element nicht mehr ohne Weiteres entfernt werden. Aus diesem Grund wird in [105] vorgeschlagen, sogenannte *Soft State Sketches* zu verwenden.

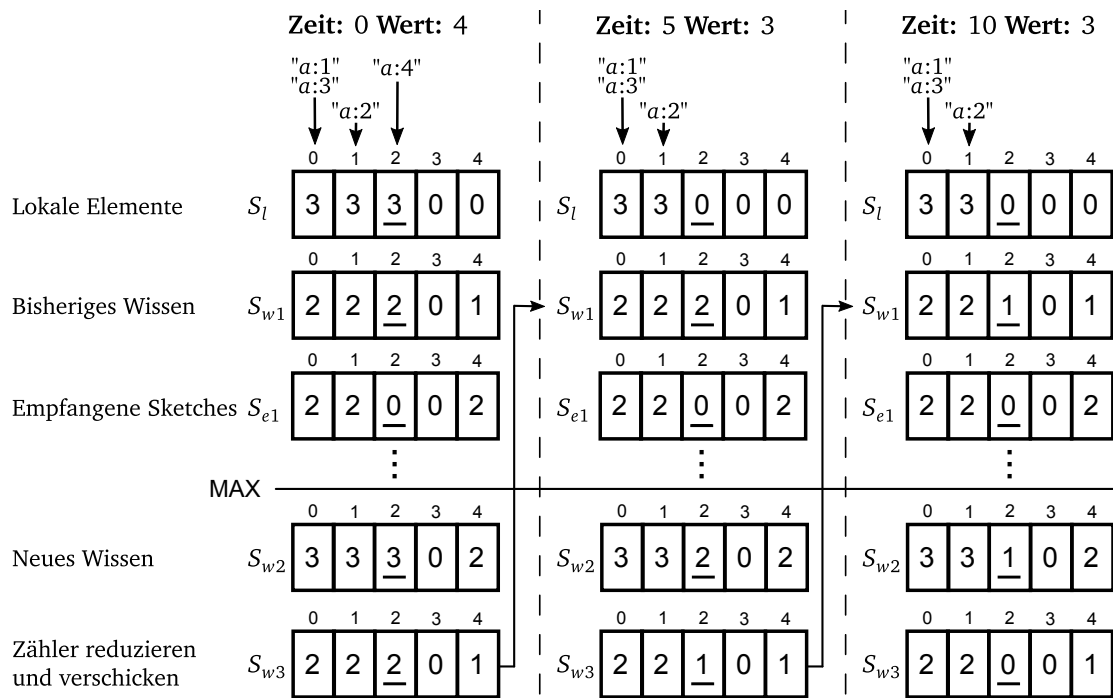
Ein Soft State Sketch der Größe  $L > 0$  besteht aus einem Zähler-Array  $S = s_0, \dots, s_{L-1}$ , wobei jeder Zähler  $N > 1$  Bit benötigt. Die grundsätzliche Idee dieser Erweiterung besteht darin, dass jeder Zähler angibt, wann das letzte Mal ein auf dieser Position abgebildetes Element eingefügt wurde. Für diesen Zweck wird jede Zahl als verbleibende Gültigkeitsdauer (Time to live, TTL) interpretiert. Dementsprechend wird beim Einfügen des Elements  $x$  der Zähler  $s_i$  mit  $i = l_1(\text{hash}(x))$  auf  $2^N - 1$  gesetzt, was der maximal codierbaren verbleibenden Gültigkeitsdauer entspricht. Als zeitliche Bezugsgröße wird das Sendeintervall der Aggregate verwendet, sodass alle Zähler  $s_i > 1$  mit  $i \in \{0, \dots, L - 1\}$  direkt vor dem Senden des Pakets um eins reduziert werden. Auf diese Weise werden nicht mehr aktualisierte Zähler schrittweise auf null reduziert. Um abzuschätzen, wie viele Elemente in einen Soft State Sketch eingefügt wurden, kann ohne Änderung die Formel 4.3 bzw. 4.4 verwendet werden. Statt durch eine bitweise OR-Operation können in diesem Ansatz mehrere Sketches durch eine positionsweise MAX-Operation der hinterlegten Gültigkeitsdauern miteinander kombiniert werden.

In Abbildung 4.9 wird der gesamte Aggregierungsalgorithmus exemplarisch aus der Sicht des Knotens  $a$  dargestellt. Zu Beginn bestimmt der Knoten den lokalen Sensorwert von vier und generiert deshalb vier Zeichenketten, die in den leeren Sketch  $S_l$  eingefügt werden. Aus vorherigen Übertragungen hat der Knoten Kenntnis von der Summe der Sensorwerte innerhalb des aktuell betrachteten Beobachtungsgebiets. Dieses lokale Wissen liegt in Form des Sketches  $S_{w1}$  vor. Des Weiteren sind dem Knoten die innerhalb des letzten Sendeintervalls empfangenen Aggregate  $S_{e1} - S_{ex}$  bekannt. Die Sketches  $S_l$ ,  $S_{w1}$  und  $S_{e1} - S_{ex}$  werden durch eine positionsweise MAX-Operation zu  $S_{w2}$  kombiniert. Vor dem Sendevorgang wird schließlich jeder Zähler, der ungleich null ist, um eins reduziert. In diesem Beispiel wird davon ausgegangen, dass sich



## 4.2 Kooperative Aggregation von orts- und zeitbezogenen Daten

weder die Topologie noch die Sensorwerte der Nachbarknoten ändern. Dementsprechend wird zu allen drei Zeitpunkten der identische Sketch  $S_{e1}$  empfangen. Allerdings reduziert sich ab Zeitpunkt 5 der lokale Sensorwert auf drei, sodass nur noch drei Elemente in den zuvor geleerten Sketch  $S_l$  eingefügt werden. Das bisher auf der zweiten Position abgebildete vierte Element ist aber weiterhin in dem lokalen Wissen  $S_{w1}$  enthalten. Da diese Position allerdings weder durch die drei verbleibenden Elemente noch durch die empfangenen Sketches aktualisiert wird, reduziert sich der entsprechende Zähler nach zwei Sendevorgängen auf null und wird dementsprechend in den zukünftigen Auswertungen nicht mehr berücksichtigt.



**Abbildung 4.9:** Das Prinzip der verbleibenden Gültigkeitsdauer: Zu Beginn werden vier Elemente generiert und in  $S_l$  eingefügt. Danach wird  $S_l$  mit dem bisherigen lokalen Wissen  $S_{w1}$  und den empfangenen Sketches  $S_{e1-x}$  zu  $S_{w2}$  kombiniert. Bevor der Sketch versendet wird, werden alle Zähler, die größer als eins sind, um eins reduziert. Ab Zeitpunkt 5 reduziert sich der lokale Sensorwert auf 3, sodass nur noch drei Elemente in  $S_l$  eingefügt werden. Das vierte Element ist zwar weiterhin in  $S_{w1}$  enthalten, da aber der entsprechende Zähler weder durch andere lokale Elemente noch durch empfangene Sketches gesetzt wird, sinkt sein Zahlenwert nach zwei Sendevorgängen auf null, wodurch das vierte Element ab diesem Zeitpunkt nicht mehr bei der Bestimmung der Kardinalität berücksichtigt wird.

Aus Gründen der Übersichtlichkeit wird im präsentierten Beispiel nur von einem Bezugsgebiet ausgegangen. In einem mehrstufigen Aggregationsschema werden allerdings pro Sendevorgang mehrere Aggregate unterschiedlicher Bezugsgebiete und Aggregationsebenen ausge-

tauscht. Dafür wird eine Auswahlstrategie benötigt, die anhand der Entfernung zwischen dem aktuellen Aufenthaltsort des Knotens und dem jeweiligen Bezugsgebiet die Priorität der lokal vorhandenen Aggregate bewertet und diejenigen mit der höchsten Priorität für den nächsten Sendevorgang auswählt. Um die Kombination von Soft State Sketches zu verdeutlichen, besteht in dem Beispiel jedes Aggregat nur aus einem Sketch. Wie in Abschnitt 4.1.1 beschrieben, ist aber die Genauigkeit eines Sketches für die meisten Anwendungen nicht ausreichend, sodass typischerweise ein Array aus Sketches verwendet wird.

### 4.2.2 Kompression

In den meisten Fällen sind 32 Bit ausreichend, um die Summe der Sensorwerte eines Gebiets zu speichern. Im Vergleich hierzu ist der Speicherbedarf von FM-Sketches beziehungsweise Soft State Sketches deutlich höher. Wenn  $M$  die Anzahl der Sketches,  $L$  die Anzahl der Zähler pro Sketch und  $N$  die Anzahl der benötigten Bit pro Zähler bezeichnet, so beträgt der gesamte Speicherbedarf  $M \cdot L \cdot N$  Bit. Die Autoren von [105, 106] verwenden in ihrer Simulationsstudie die Konfiguration  $M = 16$ ,  $L = 24$  und  $N = 5$ . Dementsprechend benötigen die Sketches innerhalb jedes Aggregats 240 Byte. Wenn auf eine Speicherung der verbleibenden Gültigkeitsdauer verzichtet wird, sinkt der benötigte Speicherplatz auf 48 Byte.

Die beim Einfügevorgang verwendete geometrische Verteilung führt dazu, dass auf der linken Seite der FM-Sketches viele Bits gesetzt werden, während die Bits auf der rechten Seite weitgehend ungesetzt bleiben. Des Weiteren werden die Elemente nahezu gleich über alle Sketches verteilt, sodass die Wahrscheinlichkeit, dass eine bestimmte Bitposition gesetzt ist, für alle Sketches identisch ist. Auf diesen Beobachtungen aufbauend, wird in [38] vorgeschlagen, eine Menge von Sketches als Matrix zu interpretieren, in der jede Zeile exakt einem Sketch entspricht. Die Idee besteht nun darin, derartige Matrizen spaltenweise zu durchlaufen und hierbei identische Bits per Lauflängencodierung zusammenzufassen. In [134] wird eine Weiterentwicklung dieser Vorgehensweise vorgeschlagen, die im Folgenden genauer erläutert wird.

### Kompression von FM-Sketches

Wie stark eine Datenstruktur verlustlos komprimiert werden kann, wird durch ihren Informationsgehalt bestimmt [144]. Sei  $n$  die Anzahl der Symbole des verwendeten Alphabets  $X = x_1, \dots, x_n$  und  $p(x_i)$  eine Funktion, die zurückgibt, mit welcher Wahrscheinlichkeit das Symbol  $x_i$  auftritt. Der mittlere Informationsgehalt wird Entropie genannt und kann mit  $H(X) = -\sum_{i=1}^n p(x_i) \cdot \log_2 p(x_i)$  berechnet werden. Wenn ein FM-Sketch auf Bitebene codiert wird, besteht das Alphabet nur aus den Symbolen 0 und 1. Allerdings ist die Wahrscheinlichkeit dafür, dass ein Bit gesetzt bzw. nicht gesetzt ist, nicht konstant, sondern von der Position innerhalb des Sketches abhängig. Unter Berücksichtigung dieser Umstände kann die Entropie eines FM-Sketches  $S = s_0, \dots, s_{L-1}$  mit

$$H_{FM}(S) = -M \sum_{i=0}^{L-1} p(s_i=0) \log_2 p(s_i=0) + p(s_i=1) \log_2 p(s_i=1) \quad (4.7)$$

bestimmt werden. Falls die Wahrscheinlichkeiten  $p(s_0 = 0), \dots, p(s_{L-1} = 0)$  bekannt sind, kann beispielsweise *Arithmetic Coding* [163] verwendet werden, um den FM-Sketch optimal zu komprimieren. Da diese Wahrscheinlichkeiten auch für die Dekompression benötigt werden, müssten sie bei einer dezentralen Anwendung mit dem komprimierten Sketch übertragen werden. Allerdings ist die hierfür zu übertragende Datenmenge im Vergleich zur Größe eines FM-Sketches unverhältnismäßig umfangreich. Dieser Umstand wird durch die Tatsache abgemildert, dass üblicherweise eine Menge von  $M$  FM-Sketches verwendet wird. In diesem Szenario verteilen sich die eingefügten Elemente nahezu gleich auf die einzelnen FM-Sketches, so dass diese spaltenweise auch die gleichen Wahrscheinlichkeiten aufweisen. Dementsprechend müssen die  $L$  Wahrscheinlichkeiten nur einmal für die Menge von FM-Sketches übertragen werden, wodurch ihr Anteil an dem gesamten Übertragungsvolumen sinkt.

[134] formuliert die Idee, die Kenntnisse über den Aufbau der FM-Sketches zu nutzen, um die Wahrscheinlichkeiten der einzelnen Symbole anhand der Anzahl der eingefügten Elemente zu berechnen. Die Wahrscheinlichkeit, dass ein Element auf das Bit  $s_i$  in einem bestimmten FM-Sketch abgebildet wird, beträgt  $2^{-(i+1)}/M$ . Wenn  $C$  unterschiedliche Elemente eingefügt wurden, dann beträgt die Wahrscheinlichkeit, dass Bit  $s_i$  in einem der FM-Sketches bisher nicht gesetzt wurde

$$p(s_i=0) = (1 - 2^{-(i+1)}/M)^C. \quad (4.8)$$

Zwar ist in dem typischen Einsatzszenario von FM-Sketches die Anzahl der eingefügten Elemente unbekannt, allerdings kann  $C$  mit Hilfe der Datenstruktur abgeschätzt werden. Anstatt die echten Wahrscheinlichkeiten für die Kompression zu verwenden, können diese mit der Formel 4.8 in Abhängigkeit von der geschätzten Anzahl der eingefügten Elemente berechnet werden. Dementsprechend muss zusätzlich zu der komprimierten Datenstruktur nur noch die geschätzte Kardinalität übertragen werden. Die Simulationsergebnisse in [134] zeigen, dass die auf diese Weise berechneten Wahrscheinlichkeiten ausreichen, um eine nahezu optimale Kompression zu erreichen. Beispielsweise benötigt eine Datenstruktur mit  $M = 256$  und  $L = 16$  unabhängig von der Anzahl der eingefügten Elemente unkomprimiert 512 Byte. Wenn bisher keine Elemente eingefügt wurden, kann die beschriebene Vorgehensweise den Speicherbedarf dieser Datenstruktur auf wenige Byte reduzieren. Mit jedem zusätzlich eingefügten Element steigt der Speicherbedarf geringfügig an und erreicht ab  $2^{10}$  Elementen das Maximum bei circa 145 Byte. Auch Konfigurationen mit nur wenigen FM-Sketches können stark komprimiert werden. Beispielsweise benötigen 16 FM-Sketches bei sonst gleich bleibenden Bedingungen nur noch etwa 12 Byte.

## Kompression von Soft State Sketches

Die in Abschnitt 4.2.1 präsentierten Soft State Sketches verwenden an jeder Position des Sketches Zähler, um die verbleibende Gültigkeitsdauer der entsprechenden Elemente zu speichern. Da hierfür jede Position mehr als ein Bit benötigt, ist es nicht möglich, die zuvor besprochene Kompressionsstrategie direkt anzuwenden. Daher wird in diesem Abschnitt eine Weiterentwicklung vorgestellt, die es ermöglicht, auch Soft State Sketches zu komprimieren.

In den vorherigen Abschnitten wurde das Konzept der verbleibenden Gültigkeitsdauer primär dazu verwendet, veraltete Informationen nach einer bestimmten Anzahl von Sendevorgängen automatisch aus den Sketches zu entfernen. Um die Kardinalität abzuschätzen, wurde hierbei analog zu den FM-Sketches Formel 4.3 herangezogen. In dieser Formel wird die Funktion  $l_b(S)$  verwendet, welche die Position des ersten Werts  $b$  innerhalb von  $S$  zurückgibt. Dabei wird die Suche von links gestartet und die Positionen werden mit 0 beginnend durchgezählt. In FM-Sketches wird  $l_0(S)$  verwendet, um die Länge der zusammenhängenden Gruppe von Einsen am Anfang des Sketches zu ermitteln. Um die Anzahl der bisher eingefügter Elemente abzuschätzen, kann  $l_0(S)$  auch bei Soft State Sketches ohne Änderung verwendet werden. Allerdings ergibt sich dabei auch die Möglichkeit, nur diejenigen Elemente zu berücksichtigen, die in einem bestimmten Umkreis eingefügt wurden. Zu diesem Zweck wird die Funktion  $l_b(S)$  dahingehend erweitert, dass sie für  $l_{<b}(S)$  die erste Position von links findet, an der ein Wert hinterlegt ist, der kleiner ist als  $b$ .

Wenn pro Zähler  $N$  Bit verwendet werden, beträgt die maximal zu hinterlegende Gültigkeitsdauer  $t_{max} = 2^N - 1$ . Da die Gültigkeitsdauer an den Sendevorgang gekoppelt ist, kann bei annähernd synchronisierten Sendevorgängen und bei Nichtberücksichtigung von Übertragungsfehlern davon ausgegangen werden, dass sich bei einem Zähler von  $s_i = t$  der nächstliegende Knoten, der ein Element auf der entsprechenden Position eingefügt hat,  $t_{max} - t$  Hops vom aktuell betrachteten Knoten entfernt befindet. Dieser Sachverhalt wird im Folgenden anhand eines Beispiels verdeutlicht. Der betrachtete Knoten  $A$  kombiniert alle von seinen Nachbarn empfangenen Sketches und erhält hierbei  $S = 66554100$ . Da alle Zähler vor dem Verschicken um eins reduziert werden und  $t_{max}$  in diesem Beispiel sieben beträgt, kann der Knoten  $A$  aus dem kombinierten Sketch  $S$  schließen, dass die Zähler  $s_0$  und  $s_1$  von mindestens einem direkt benachbarten Knoten aktualisiert wurden. Die in den Zähler  $s_2$  und  $s_3$  gespeicherte verbleibende Gültigkeitsdauer beträgt fünf, woraus folgt, dass sich diejenigen Knoten, die diese Zähler zuletzt aktualisiert haben, exakt zwei Hops von  $A$  entfernt befinden. Auf dieser Beobachtung aufbauend kann die Anzahl derjenigen Elemente, die in einem Umkreis von  $0 \leq h < t_{max}$  Hops eingefügt wurden, mit der folgenden Funktion abgeschätzt werden:

$$C_U(S_0, \dots, S_{M-1}, h) = \frac{M}{p} \cdot 2^{\sum_{j=0}^{M-1} l_{<t_{max}-h}(S_j)/M}. \quad (4.9)$$

Diese Bereichsabschätzung kann auch genutzt werden, um die Anzahl derjenigen Elemente zu überschlagen, die in einer Entfernung von exakt  $h$  Hops zuletzt aktualisiert wurden. Dafür wird, wie in der folgenden Formel dargestellt, die Anzahl derjenigen Elemente, die im Umkreis

von  $h - 1$  Hops eingefügt wurden, von der Anzahl derjenigen Elemente subtrahiert, die im Umkreis von  $h$  Hops eingefügt wurden.

$$C_E(S_0, \dots, S_{M-1}, h) = \begin{cases} C_U(S_0, \dots, S_{M-1}, h) - C_U(S_0, \dots, S_{M-1}, h-1) & \text{für } 0 < h < t_{\max} \\ C_U(S_0, \dots, S_{M-1}, 0) & \text{für } h = 0 \end{cases} \quad (4.10)$$

Wie im vorherigen Abschnitt erläutert, beträgt die Wahrscheinlichkeit, dass nach  $C$  Einfügevorgängen kein Element auf der Position  $s_i$  abgebildet wurde,  $(1 - 2^{-(i+1)}/M)^C$  (siehe Formel 4.8). Dementsprechend kann die Wahrscheinlichkeit, dass nach  $C$  Einfügevorgängen mindestens ein Element auf die Position  $s_i$  abgebildet wurde, mit

$$p(s_i \neq 0, C) = 1 - (1 - 2^{-(i+1)}/M)^C \quad (4.11)$$

bestimmt werden. Das Symbol  $t \in \{1, \dots, t_{\max} - 1\}$  kann nur auftreten, wenn die entsprechende Position von einem exakt  $h = t_{\max} - t$  Hops weit entfernten Knoten aktualisiert und gleichzeitig die Position von keinem näher liegenden Knoten überschrieben wurde. Gemäß dieser Beobachtung und unter Berücksichtigung der beiden Sonderfälle kann die Eintrittswahrscheinlichkeit aller Symbole folgendermaßen abgeschätzt werden:

$$p(s_i = t) = \begin{cases} p(s_i = 0, C_U(S, t_{\max} - 1)) & \text{für } t = 0 \\ p(s_i \neq 0, C_E(S, t_{\max} - t)) \cdot p(s_i = 0, C_U(S, t_{\max} - t - 1)) & \text{für } 0 < t < t_{\max} \\ p(s_i \neq 0, C_E(S, 0)) & \text{für } t = t_{\max} \end{cases} \quad (4.12)$$

Die bisher präsentierten Formeln beziehen sich auf lokale Sketches, die aus den eigenen Beobachtungen und den empfangenen Sketches der benachbarten Knoten zusammengesetzt werden. Bezogen auf das in Abbildung 4.9 dargestellte Aggregierungsschema bedeutet dies, dass sich alle Formeln auf  $S_{w2}$  beziehen. Wenn ein Knoten die Sketches zu seinen Nachbarn überträgt, muss die verbleibende Gültigkeitsdauer der neuen Situation angepasst werden. Ob hierfür die Zähler vom Sender oder vom Empfänger um eins reduziert werden, ist algorithmisch gleichwertig. Die zuerst genannte Variante hat aber den Vorteil, dass die zu übertragenden Sketches nicht mehr  $t_{\max}$  enthalten können und sich somit ihr Informationsgehalt reduziert. Bei einer geschickten Wahl der Konfigurationsparameter kann so die Anzahl der pro Zähler benötigten Bits um eins reduziert werden. Aus diesem Grund wird analog zu dem zuvor besprochenen Aggregierungsschema auch in den folgenden Abschnitten immer davon ausgegangen, dass die Zähler unmittelbar vor dem Versenden reduziert werden.

Mit Hilfe der Formel 4.12 können die Wahrscheinlichkeiten bestimmt werden, mit denen die Symbole  $0, \dots, 2^N - 1$  an den Positionen  $0, \dots, L - 1$  der  $M$  Sketches auftreten können. Diese Wahrscheinlichkeiten können dazu verwendet werden, um Soft State Sketches mit Arithmetic Coding nahezu optimal zu komprimieren. Damit die Sketches allerdings wieder dekomprimiert

miert werden können, müssen zusätzlich zu den komprimierten Daten die Zahlen  $C_U(S, h)$  für  $h \in \{0, \dots, t_{max} - 1\}$  bekannt sein. Hierbei ist im Fall der Kompression der zu übertragenden Sketches zu beachten, dass sich durch die Reduktion aller Zähler auch  $t_{max}$  um eins reduziert. Des Weiteren entspricht der zu komprimierende Sketch dann der Sicht des Empfängers, sodass sich die Umrechnung der verbleibenden Gültigkeitsdauer in Hops ebenfalls um eins verschiebt. Dementsprechend müssen für die Kompression die jeweiligen Formeln angepasst werden.

Für die Übertragung der Zahlen  $C_U(S, h)$  mit  $h \in \{0, \dots, t_{max} - 1\}$  sind pro Zahl meist 16 Bit Speicherplatz ausreichend. Dementsprechend müssen pro Aggregat  $2 \cdot t_{max}$  Byte zusätzlich übertragen werden. Ob dieses Datenvolumen in Kombination mit den komprimierten Daten noch deutlich unter der unkomprimierten Größe der Sketches liegt, hängt von der jeweiligen Konfiguration ab. In dem weiter oben genannten Szenario mit  $M = 16$ ,  $L = 24$  und  $N = 5$  würden die Zahlen 62 Byte benötigen. Allerdings ist in Anbetracht der zu erwartenden Einsparung durch die Kompression dennoch davon auszugehen, dass der insgesamt benötigte Speicherplatz noch deutlich unter den unkomprimiert benötigten 240 Byte liegt. Im Vergleich zu der Kompression vom FM-Sketches wäre die Einsparung in diesem Beispiel aber wesentlich geringer.

Da die Anzahl der zu übertragenden Zahlen ausschließlich von  $N$  abhängt, ist das vorgestellte Schema besonders geeignet, wenn das Aggregierungsgebiet nur wenige Hops umfasst. Des Weiteren sind Szenarien von Vorteil, in denen eine höhere Aggregierungsgenauigkeit benötigt wird und dementsprechend deutlich mehr als die oben genannten 16 Sketches eingesetzt werden. Wenn beispielsweise die Konfiguration  $M = 128$ ,  $L = 16$  und  $N = 3$  verwendet wird, benötigt jedes Aggregat unkomprimiert 768 Byte. In diesem Fall werden aber nur  $2 \cdot (2^3 - 1) = 14$  Byte für die Zahlen benötigt, wodurch die zu erwartende Einsparung durch die Kompression deutlich größer ausfallen dürfte als in dem vorherigen Beispiel.

Damit ein Datenstrom mit Hilfe der Entropiecodierung komprimiert und dekomprimiert werden kann, müssen die Eintrittswahrscheinlichkeiten aller Symbole bekannt sein. Bei kleineren Datenmengen kann die Übertragung dieser Wahrscheinlichkeiten die Kompressionsersparnis schnell übersteigen. Aus diesem Grund wird unter anderem in [163] ein Ansatz vorgeschlagen, in welchem die Wahrscheinlichkeiten während der Kompression anhand der zuvor komprimierten Zeichen abgeschätzt werden. Während der Dekompression wird der identische Algorithmus ausgeführt, sodass sichergestellt wird, dass auf beiden Seiten für jedes Zeichen exakt die gleichen Wahrscheinlichkeiten verwendet werden. Allerdings gehen diese sogenannten dynamischen Modelle von einer Gleichverteilung der Symbole über den zu komprimierenden Datenstrom aus und können somit nicht unmittelbar für die Kompression von Sketches eingesetzt werden. Die Autoren von [106] haben daher eine Formel hergeleitet, mit der die Wahrscheinlichkeit eines bestimmten Symbols in der Spalte  $i - 1$  abhängig von seiner Wahrscheinlichkeit in der Spalte  $i$  berechnet werden kann. In diesem Ansatz startet das dynamische Modell in der letzten Spalte der als Matrix interpretierten Sketches und bestimmt anhand der dort vorhandenen Symbole die entsprechenden Wahrscheinlichkeiten. Nachdem die Spalte abgearbeitet wurde, werden die ermittelten Wahrscheinlichkeiten mit Hilfe der zuvor erwähnten Formel

auf die neue Spalte umgerechnet. Dieser Vorgang wird so lange wiederholt, bis alle Spalten durchlaufen wurden. In [106] wird diese Strategie verwendet, um die innerhalb einer Simulationsstudie verschickten Sketches zu komprimieren. Dabei konnte bei einer Konfiguration von  $M = 16$ ,  $L = 24$  und  $N = 5$  der Speicherbedarf in den meisten Fällen auf 10-50 Byte reduziert werden. Wie viele Elemente die Sketches zum Zeitpunkt der Kompression enthalten haben oder wie sich die Resultate im Verhältnis zur jeweiligen Entropie verhalten, wird nicht angegeben. Da dieses Verfahren erst nach dem Abschluss der im folgenden Abschnitt vorgestellten empirischen Evaluation publiziert wurde, wird es im Folgenden nicht weiter berücksichtigt. Unter der Voraussetzung, dass das dynamische Modell die realen Wahrscheinlichkeiten der Symbole schon nach wenigen Eingabezeichen während der Kompression akkurat schätzen kann, können die folgenden Resultate allerdings teilweise auf diesen Ansatz übertragen werden.

### 4.2.3 Simulationsumgebung

Die zuvor beschriebenen Soft State Sketches und das Prinzip der verbleibenden Gültigkeitsdauer sind geeignet, um Sensordaten dezentral zu aggregieren. In der folgenden Simulationsstudie wird untersucht, mit welcher Genauigkeit die Aggregierungsfunktionen COUNT, SUM und AVG mit Soft State Sketches bestimmt werden können. Dabei wird ebenfalls die Genauigkeit der Bereichsabfragen gemäß Formel 4.10 evaluiert. Des Weiteren wird untersucht, wie viel Speicherplatz die komprimierten Datenstrukturen in Abhängigkeit von ihrer Konfiguration benötigen. Im Unterschied zur Evaluation der FM-Sketches werden hierfür aber eine Netzwerktopologie und ein Datenmodell benötigt. Anhand der Netzwerktopologie wird festgestellt, mit welcher verbleibenden Gültigkeitsdauer welches Element von welchen Knoten empfangen werden kann. Wie viele Elemente die einzelnen Knoten dabei in die Sketches einfügen, wird wiederum vom Datenmodell festgelegt.

In Abschnitt 5.4 wird eine Feldstudie vorgestellt, in deren Rahmen alle zugänglichen Straßen der Mannheimer Innenstadt mit einem Fahrzeug abgefahren wurde. Das Fahrzeug war mit mehreren WLAN-Empfängern ausgestattet und zeichnete während der Fahrt alle empfangenen WLAN-Pakete auf. Da gleichzeitig auch die jeweils aktuelle GPS-Position aufgezeichnet wurde, konnten nach der Messung die Positionen der vorhandenen Access Points abgeschätzt werden. Anhand dieser Standorte und einer statistischen Auswertung der Messdaten konnte in etwa abgeschätzt werden, welcher Access Point von welchen anderen Access Points Pakete empfangen kann. Die dabei ermittelten Knoten und Kanten werden in der folgenden Simulationsstudie als Netzwerktopologie verwendet.

Welche Sensorwerte die 3797 Knoten jeweils in die Sketches einfügen, wird zu Beginn eines Simulationsdurchlaufs festgelegt. Da der verwendete Netzwerkgraph recht dicht ist und somit die Summe der Sensorwerte im Beobachtungsgebiet schnell steigt, wird in der Simulation der Bereich der Sensorwerte auf  $0, \dots, 255$  beschränkt. Welcher Knoten welchen Sensorwert aus diesem Wertebereich verwendet, wird zufällig ausgewählt. Dabei kann der Simulator sowohl gleich verteilte als auch normal verteilte Zufallszahlen generieren. Die Unterschiede zwischen den Simulationsergebnissen der beiden Verteilungen sind allerdings sehr gering, sodass hier



nur die Graphen der Durchläufe mit gleich verteilten Sensorwerten präsentiert werden. Neben den Sensorwerten generiert der Simulator für jeden Knoten noch einen zufälligen, aber eindeutigen Identifikator, der für das Generieren der Zeichenketten während des Einfügevorgangs verwendet wird.

Gemäß dem verwendeten verteilten Algorithmus summiert jeder Knoten die Sensorwerte aller Knoten innerhalb eines Radius von  $t_{max} - 1$  Hops. Um diesen Datenaustausch zu simulieren, werden in die Sketches des Knotens  $v$  die Sensorwerte aller Knoten  $m \in NB_{v, t_{max}-1}$  mit der jeweiligen Gültigkeitsdauer  $t_{max} - kwl(v, n)$  eingefügt. Wie in Abschnitt 3.4.1 gibt hierbei die Funktion  $NB_{v, x}$  die Menge aller Knoten innerhalb eines Umkreises von  $x$  Hops um den Knoten  $v$  zurück. Aus diesem Abschnitt wurde ebenfalls die Funktion  $kwl(v, n)$  übernommen, welche die Länge des kürzesten Wegs zwischen den Knoten  $v$  und  $n$  bestimmt. Nach dem Einfügen der Elemente wird der durchschnittliche relative Fehler zwischen den realen Summen und den Abschätzungen anhand der entsprechenden Sketches berechnet. Zusätzlich wird auch der relative Fehler bezüglich der Bereichsabschätzungen gemäß Formel 4.10 bestimmt. Analog zu der beschriebenen Vorgehensweise wird anhand eines zweiten Sketches evaluiert, mit welcher Genauigkeit die Anzahl der Knoten im Beobachtungsgebiet bestimmt werden kann. Die Summe und die Anzahl der Knoten werden des Weiteren dazu genutzt, zu evaluieren, wie exakt der durchschnittliche Sensorwert mit Hilfe von Sketches abgeschätzt werden kann. Neben der Genauigkeit wird in der Simulation auch der benötigte Speicherplatz evaluiert, sodass die Sketches abschließend mit dem oben beschriebenen Verfahren komprimiert werden. Da nicht der Speicherbedarf der lokal vorliegenden Sketches, sondern der über das Netzwerk ausgetauschten Datenstrukturen von Interesse ist, werden vor der Komprimierung alle Zähler, die größer als eins sind, um eins reduziert. Hierdurch reduziert sich auch das bei der Komprimierung berücksichtigte  $t_{max}$  um eins. Der beschriebene Ablauf ist in Algorithmus 4.1 zusammengefasst. Alle im Folgenden präsentierten Graphen basieren auf den gemittelten Ergebnissen von jeweils hundert Simulationsdurchläufen.

Um die Kardinalität abzuschätzen, verwendet der Simulator Formel 4.3 und das in Abschnitt 4.1.4 vorgestellte Verfahren. Damit gewährleistet werden kann, dass in den Sketches mehr Elemente als die empfohlenen  $15 \cdot M$  enthalten sind, wird die Anzahl der einzufügenden Elemente mit einem vorgegebenen Faktor multipliziert, durch den beim Abschätzung der Kardinalität wieder dividiert wird. Diese Vorgehensweise ist insbesondere für die COUNT-Funktion notwendig. Wenn beispielsweise 512 Sketches verwendet werden, müssten sich ansonsten im Beobachtungsgebiet mindestens 7680 Knoten befinden, um die empfohlene Anzahl von Elementen pro Sketch zu erreichen. Gleichzeitig sollte sichergestellt werden, dass bei der verwendeten Konfiguration von  $L = 16$  die maximale Anzahl von Elementen pro Sketch nicht überschritten wird. Aus diesem Grund wählt der Simulator abhängig von der jeweiligen Konfiguration einen geeigneten Multiplikator aus. Aufbauend auf der Annahme, dass sich mindestens zwölf Knoten im Beobachtungsgebiet befinden, wird  $f_c = MIN(M \cdot 15/12, 400)$  als Multiplikator für die COUNT-Funktion verwendet. Da für das Generieren der Sensorwerte gleich verteilte Zufallszahlen aus dem Wertebereich 0 bis 255 herangezogen werden, beträgt der im



---

**Algorithmus 4.1** : Algorithmische Darstellung eines Simulationsdurchlaufs.

---

```

s := generiereSensorWerteArray();
i := generiereIdentifikatorenArray();
for v ∈ V do
    count := erzeugeLeeresArray();
    sum := erzeugeLeeresArray();
    cs := erzeugeCountSketch();
    ss := erzeugeSumSketch();
    for n ∈ NBv, tmax-1 do
        hopDistanz := kwl(v, n);
        gültigkeit := tmax − hopDistanz;
        count[hopDistanz]++;
        cs.einfügen(1, i[n], gültigkeit);
        sum[hopDistanz] += s[n];
        ss.einfügen(s[n], i[n], gültigkeit);
    relativerFehlerCOUNT(count, cs);
    relativerFehlerSUM(sum, ss);
    relativerFehlerAVG(count, sum, cs, ss);
    for t ∈ [1, tmax − 1] do
        relativerFehlerCOUNT(count, cs, t);
        relativerFehlerSUM(sum, ss, t);
        relativerFehlerAVG(count, sum, cs, ss, t);
    bestimmeEntropie(cs, ss);
    komprimiere(cs, ss);

```

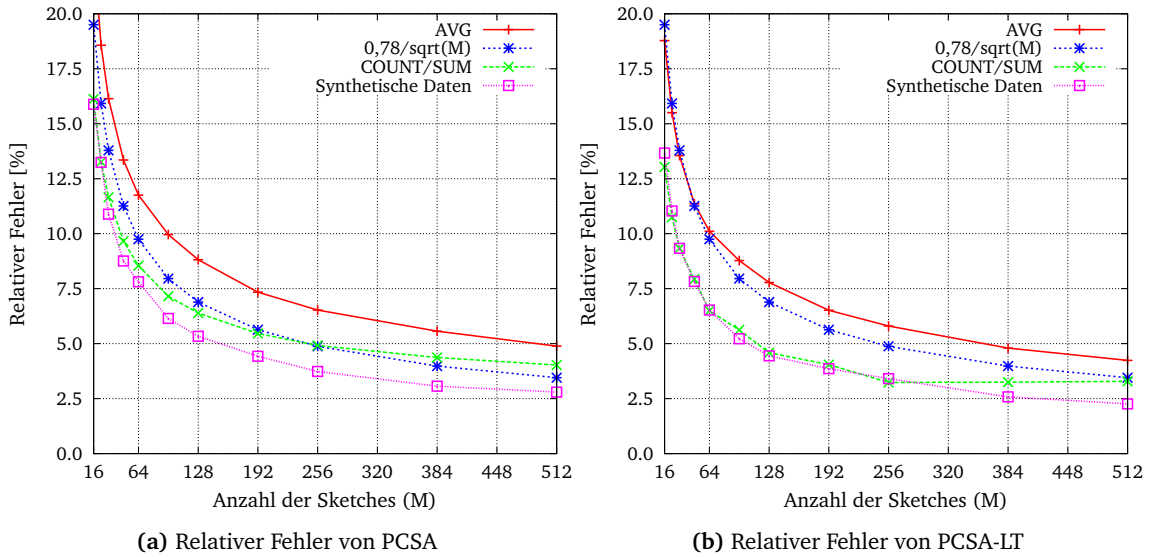
---

Durchschnitt zu erwartende Sensorwert 127,5. Dementsprechend beträgt die durchschnittliche Summe der Sensorwerte bei der angenommenen Mindestanzahl von Knoten im Beobachtungsgebiet 1530. Aus diesem Grund wird von dem Simulator für die Sketches, welche die Summe der Sensorwerte speichern, der Multiplikator  $f_s = \text{MAX}(1, M \cdot 15/1530)$  verwendet. Folglich kann – ausgehend von dem zu erwartenden durchschnittlichen Sensorwert – mit  $M = 16$  die Summe von circa 500 Knoten ermittelt werden. Zwar wird bei einem Beobachtungsradius von acht Hops diese Anzahl in der Simulation deutlich überschritten, da sich allerdings die Genauigkeit der Schätzung bei einem Überschreiten des Grenzwerts von 4000 Elementen pro Sketch nur langsam verschlechtert, ist diese einer Unterschreitung der Mindestanzahl vorzuziehen.

### 4.2.4 Simulationsergebnisse

Ein wichtiges Ergebnis der Simulation ist die erreichte Genauigkeit der untersuchten Aggregierungsfunktionen. Abbildung 4.10 zeigt den durchschnittlichen Fehler für die Funktionen COUNT, SUM und AVG. Auf der linken Seite sind die Ergebnisse von PCSA und auf der rechten

die der in Abschnitt 4.1.4 vorgestellten Abschätzungsstrategie zu sehen. Zu Vergleichszwecken sind ebenfalls die Ergebnisse der vorherigen Simulationsstudie mit synthetisch generierten Werten eingezeichnet.



**Abbildung 4.10:** Der relative Fehler von PCSA und PCSA-LT für unterschiedliche Aggregierungsfunktionen im Vergleich zu den Simulationsergebnissen mit synthetischen Daten.

Die Berechnung von COUNT und SUM entspricht direkt der Abschätzung der Kardinalität der jeweiligen Sketches. Durch das Verwenden von nicht ganzzahligen Multiplikatoren entstehen Rundungsfehler; daher liegt in den Simulationsergebnissen der relative Fehler von SUM geringfügig über dem von COUNT. Da aber beide Funktionen auf den gleichen Algorithmen basieren und dementsprechend auch die gleiche Genauigkeit erreichen können, ist in der Abbildung nur der exemplarische Verlauf von COUNT dargestellt.

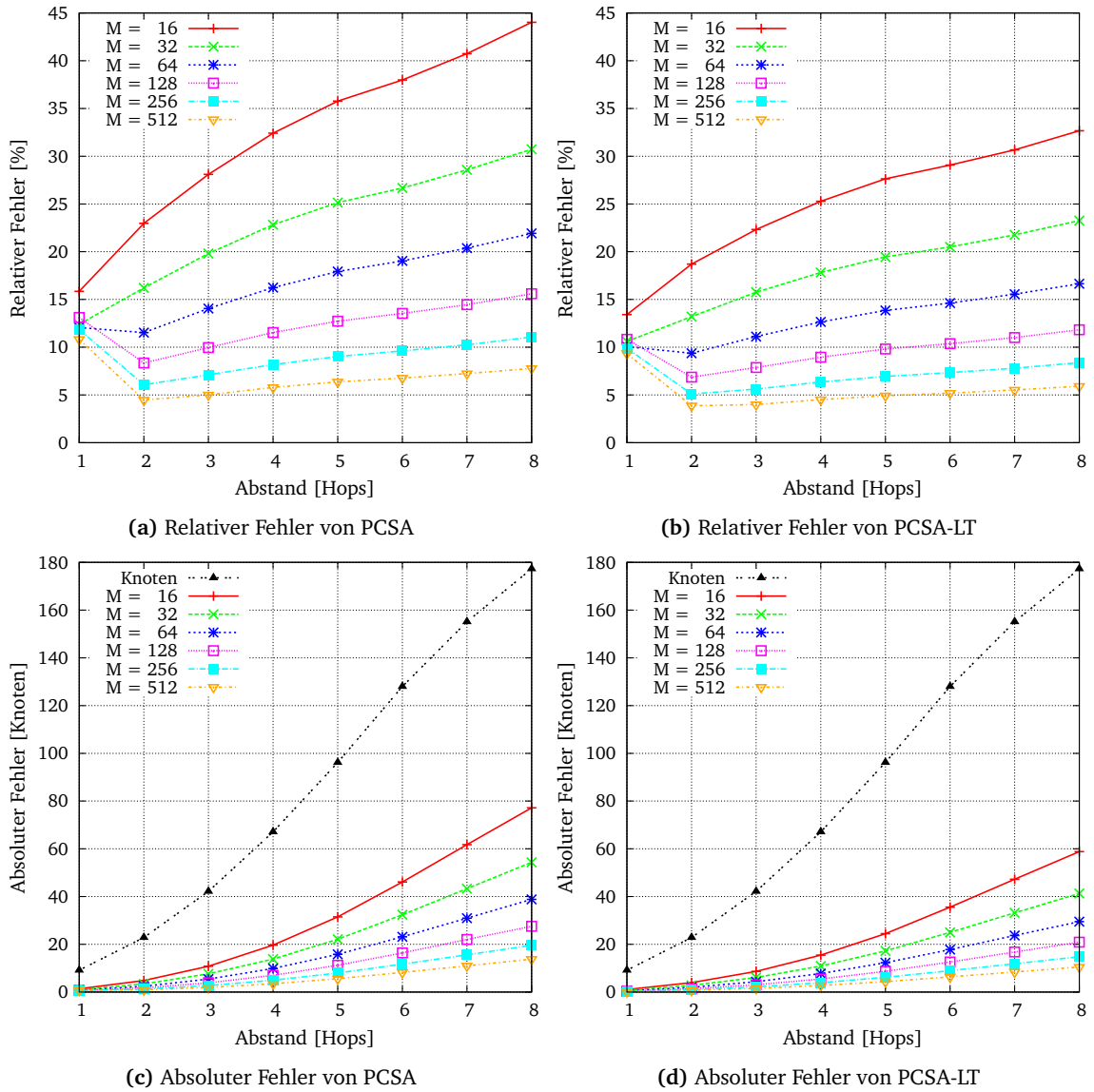
Der durchschnittliche relative Fehler von PCSA liegt deutlich über den Ergebnissen der vorherigen Simulationsstudie und nähert sich in etwa dem in der Erstpublikation angegebenen Wert von  $0,78/\sqrt{M}$  an. So erhöht sich beispielsweise der relative Fehler bei  $M = 64$  von circa 7,9 % auf circa 8,6 %. Im Unterschied dazu entsprechen die ermittelten Fehler von PCSA-LT – zumindest für  $M \leq 256$  – nahezu den Ergebnissen mit synthetisch generierten Elementen. Dadurch vergrößert sich der Genauigkeitsunterschied zwischen beiden Verfahren deutlich. Beispielsweise beträgt der durchschnittliche relative Fehler von PCSA-LT bei  $M = 64$  etwa 6,5 % und liegt somit etwa um 2,1 Prozentpunkte unter den Ergebnissen von PCSA.

Die Funktion AVG entspricht der Division der Summe aller Sensorwerte durch die Anzahl aller Knoten im Beobachtungsgebiet. Da diese Berechnung anhand von zwei fehlerbehafteten Abschätzungen erfolgt, liegt der zu erwartende relative Fehler von AVG über den Ergebnissen von COUNT und SUM. Beispielsweise beträgt bei  $M = 64$  der relative Fehler von PCSA circa 11,8 % und liegt somit etwa 3,2 Prozentpunkte über den Resultaten der beiden verwendeten

Aggregierungsfunktionen. Bei PCSA-LT steigt der durchschnittliche relative Fehler zwar um rund 3,5 Prozentpunkte, liegt aber mit 10,0 % weiterhin deutlich unter den Ergebnissen von PCSA.

Wie in Abschnitt 4.2.2 erläutert, können mit Soft State Sketches die Aggregierungsfunktionen nicht nur auf das gesamte Beobachtungsgebiet angewendet werden, sondern auch auf diejenigen Knoten, die sich in einem bestimmten Abstand zum jeweils betrachteten Knoten entfernt befinden. Neben den Anwendungen zur verteilten Datenaggregierung ist Formel 4.10 auch für solche Anwendungen von Bedeutung, die Informationen über die Netzwerkdichte in ihrer Umgebung benötigen, um beispielsweise die Senderate der Pakete festzulegen. Abbildung 4.11 zeigt sowohl den relativen als auch den absoluten Fehler dieser Abschätzungen in Abhängigkeit von der Anzahl der Sketches und dem Abstand in Hops. Generell sind alle Fehler aufgrund der zusammengesetzten Berechnungsweise deutlich größer als die jeweiligen Ergebnisse aus dem gesamten Beobachtungsgebiet. Da die Anzahl der direkt benachbarten Knoten stark schwankt und so bei vielen Knoten die empfohlene Mindestanzahl von Elementen in den Sketches nicht erreicht wird, liegt der Fehler insbesondere für  $M \geq 128$  zu Beginn besonders hoch. Ab einem Abstand von zwei Hops beeinflusst zwar die Mindestanzahl die Ergebnisse nicht mehr signifikant, allerdings steigt aufgrund der Berechnungsweise der Fehler mit jedem weiteren Hop an. Wenn beispielsweise 64 Sketches verwendet werden, betragen die relativen Fehler von PCSA und PCSA-LT bei einem Abstand von einem Hop 12,1 % beziehungsweise 10,1 %. Bei dieser Konfiguration steigt der Fehler abhängig vom Abstand auf maximal 21,9 % beziehungsweise 16,6 % an. In der zweiten Zeile der Abbildung wurden die Ergebnisse auch als absoluter Fehler in Relation zur durchschnittlichen Anzahl von Knoten im jeweiligen Abstand eingezeichnet.

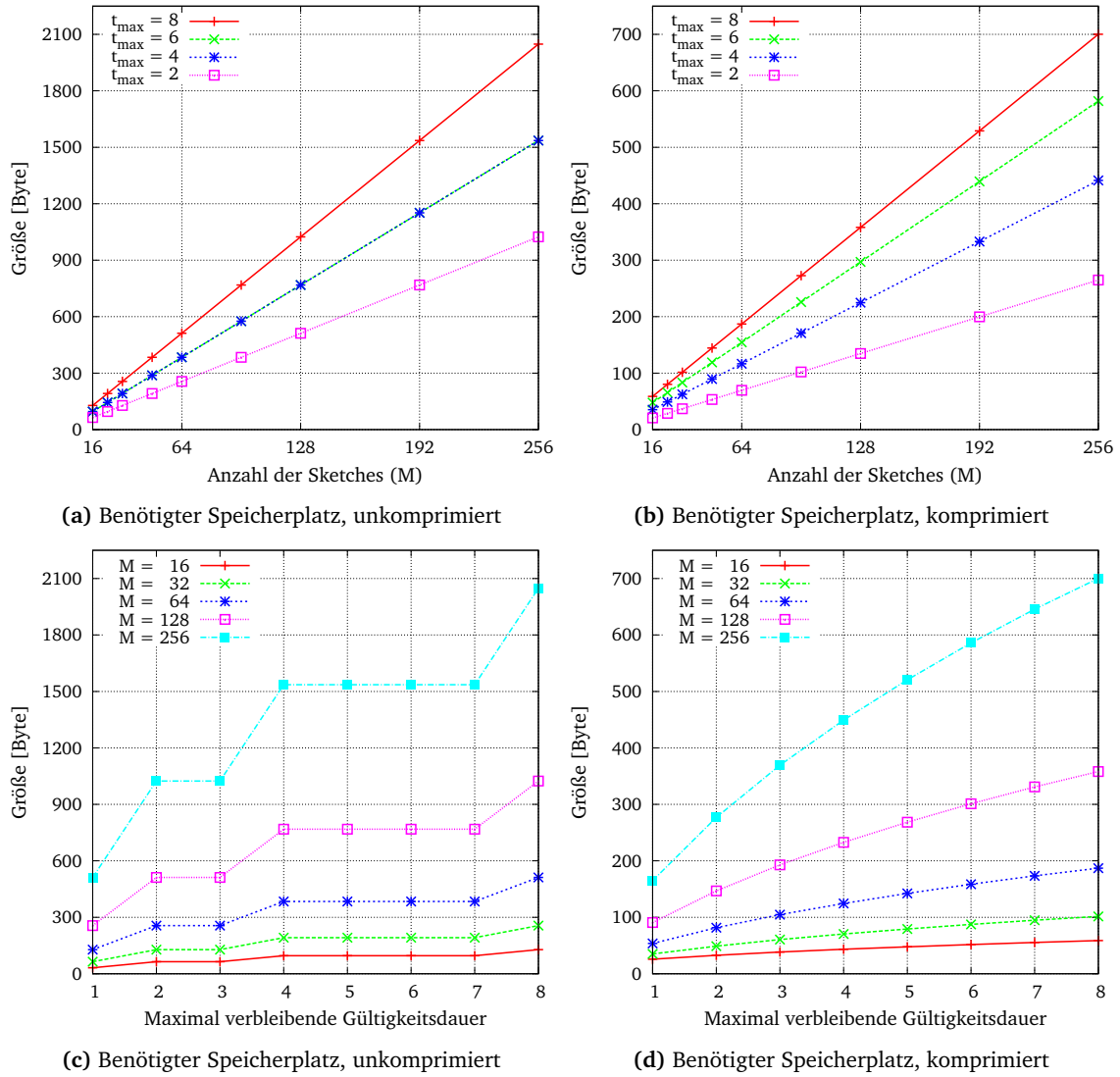
Neben der Genauigkeit einer bestimmten Konfiguration ist auch der jeweils benötigte Speicherplatz von hohem Interesse. Aus diesem Grund wurde in Abbildung 4.12 der benötigte Speicherplatz in Abhängigkeit von der Anzahl der Sketches und der maximal verbleibenden Gültigkeitsdauer eingezeichnet. Alle angegebenen Werte beziehen sich auf eine einzelne Datenstruktur. Daher muss der Speicherbedarf verdoppelt werden, wenn sowohl COUNT als auch SUM bestimmt werden sollen. Während auf der linken Seite der Speicherbedarf der unkomprimierten Sketches eingezeichnet wurde, basieren die auf der rechten Seite dargestellten Ergebnisse auf der in Abschnitt 4.2.2 vorgestellten Kompressionsstrategie. Die angegebenen Werte enthalten bereits die zur Dekomprimierung benötigten zusätzlichen  $2 \cdot t_{max}$  Byte. Bei gleich bleibender maximaler Gültigkeitsdauer steigt der Speicherbedarf sowohl unkomprimiert als auch komprimiert linear mit der Anzahl der verwendeten Sketches an. Wenn die ausschließlich von der maximalen Gültigkeitsdauer abhängigen zusätzlichen Werte außer Acht gelassen werden, ist das Verhältnis zwischen der unkomprimierten und der komprimierten Größe nahezu konstant. Wenn beispielsweise eine maximale Gültigkeitsdauer von 8 gespeichert werden soll, benötigen 64 Sketches mit jeweils 16 Zählern insgesamt 512 Byte. Durch die Kompression kann der Speicherbedarf im Durchschnitt auf circa 171 Byte reduziert werden, was in etwa einem Drittel der ursprünglichen Größe entspricht. Diese prozentuale Reduktion ergibt sich



**Abbildung 4.11:** Relativer und absoluter Fehler von PCSA und PCSA-LT bezüglich der Abschätzung der Anzahl der Nachbarn in einem bestimmten Abstand.

## 4.2 Kooperative Aggregation von orts- und zeitbezogenen Daten

auch bei einer anderen Anzahl von Sketches. Beispielsweise sinkt der Speicherbedarf durch die Kompression bei  $M = 256$  von 2048 Byte auf rund 684 Byte.



**Abbildung 4.12:** Der von Soft State Sketches benötigte Speicherplatz in Abhängigkeit von der Anzahl der Sketches und der maximal verbleibenden Gültigkeitsdauer.

Bei dieser Konfiguration benötigt jeder Zähler innerhalb eines Sketches vier Bit. Das Symbol 0 codiert den Fakt, dass bisher kein eingefügtes Element auf der jeweiligen Position abgebildet wurde. Die Symbole 1 bis 8 entsprechen der jeweils verbleibenden Gültigkeitsdauer. Die Zahlen 9 bis 15 bleiben bei dieser Konfiguration ungenutzt, sodass der Speicherplatz nicht effektiv ausgelastet ist. Doch auch wenn alle möglichen Bitkombinationen genutzt werden, ist die Speicherplatzeinsparung durch die Kompression enorm. So kann beispielsweise bei  $t_{max} = 7$  oder

bei  $t_{max} = 3$  der Speicherbedarf unabhängig von der Sketchanzahl auf circa 41 % beziehungsweise 34 % reduziert werden.

In der zweiten Zeile von Abbildung 4.12 wird der benötigte Speicherplatz in Abhängigkeit von der maximal verbleibenden Gültigkeitsdauer dargestellt. Basierend auf der verwendeten Topologie sinkt der zusätzlich benötigte Speicherplatz der komprimierten Sketches mit jeder weiteren Vergrößerung des Beobachtungsgebiets. Dies begründet sich aus dem Umstand, dass ein Zähler in den kombinierten Sketches aller empfangenen Datensätze nur den Wert  $0 < X < t_{max}$  aufweisen kann, wenn mindestens ein auf diese Position abgebildetes Element auf einem Knoten mit einem Abstand von  $t_{max} - X$  Hops eingefügt wurde und gleichzeitig kein anderer Knoten in einem Radius von  $t_{max} - X - 1$  Hops ein Element eingefügt hat, welches ebenfalls auf die gleiche Position abgebildet wurde. Dementsprechend steigt bei einer Vergrößerung des Beobachtungsgebiets auch die Wahrscheinlichkeit, dass ein auf einem Randknoten gesetzter Zähler von einem näher liegenden Knoten überschrieben wird. Auf diese Weise verursachen die zusätzlichen Knoten bei einer Vergrößerung des Beobachtungsgebiets weniger zusätzliche Entropie als die bisherigen Symbole im Durchschnitt, sodass es zu dem in der Abbildung dargestellten unterlinearen Kurvenverlauf kommt. Bei der angegebenen Berechnung ist zu berücksichtigen, dass es sich hierbei um die Sichtweise des zu komprimierenden Sketches handelt und sich deshalb  $t_{max}$  auf die zu übertragenden Daten nach der Reduktion aller Zähler bezieht (vgl. Abbildung 4.9 und Abschnitt 4.2.2).

### 4.3 Bloomfilter-Maps

Die Ergebnisse der im vorherigen Abschnitt beschriebenen Simulationsstudie zeigen, dass Soft State Sketches und die darauf aufbauende Verbreitungsstrategie für die dezentrale Aggregation von Sensordaten geeignet sind. Für zahlreiche Anwendungen sind die erreichte Genauigkeit und der jeweils benötigte Speicherbedarf ausreichend. Beispielsweise kann mit  $M = 128$  und  $t_{max} = 8$  bei einer COUNT- oder SUM-Funktion eine Genauigkeit von circa 6,9 % beziehungsweise mit der verbesserten Abschätzung von circa 4,5 %, erreicht werden. Hierbei benötigen die komprimierten Sketches für COUNT und SUM jeweils etwa 350 Byte. Da der Speicherplatz mit der Vergrößerung des Beobachtungsgebiets leicht unterlinear steigt, kann bei gleich bleibender Anzahl der Sketches sogar die Datenstruktur für ein deutlich größeres Beobachtungsgebiet noch in einem einzelnen Datenpaket übertragen werden.

Grundsätzliche Nachteile dieses Ansatzes sind allerdings die komplexe Konfiguration und die hohe Anzahl von Nebenbedingungen, die berücksichtigt werden müssen. So muss zum Beispiel sichergestellt werden, dass immer die jeweilige Mindestanzahl von Elementen eingefügt wird, damit die Schätzung auch zuverlässige Ergebnisse liefert. Dieser Umstand führt dazu, dass die zu erwartende Anzahl der Teilnehmer in etwa bekannt sein sollte. Soll beispielsweise die Anzahl der Knoten mit einer bestimmten Eigenschaft bestimmt werden, so müssten sich bei  $M = 16$  circa 240 entsprechende Knoten in der Umgebung befinden, um die in der Publikation geforderte Mindestanzahl von  $M \cdot 15$  zu erreichen. Die Ergebnisse in Abschnitt 4.1.2

zeigen aber, dass sich der relative Fehler schon weit früher stabilisiert und somit in der Praxis auch eine deutlich niedrigere Mindestanzahl ausreichend ist. Des Weiteren kann durch die Verwendung der modifizierten Abschätzung gemäß Formel 4.4 die Mindestanzahl nochmals reduziert werden. Dennoch ist die Mindestanzahl der Teilnehmer von  $M$  abhängig und muss dementsprechend vorab bei der Konfiguration berücksichtigt werden. Wenn beispielsweise eine höhere Genauigkeit benötigt und deshalb  $M = 256$  verwendet wird, müssen sich auch mit den genannten Verbesserungen mindestens 800 Teilnehmerknoten im Beobachtungsgebiet befinden, um im Durchschnitt den entsprechenden relativen Fehler zu erreichen. Dieses Problem kann gelöst werden, indem jeder Knoten eine konstante Zahl von Elementen in die Sketchmenge einfügt. In dem zuletzt genannten Beispiel werden bei einem Multiplikator von 50 nur noch 16 Knoten in der Umgebung benötigt. Dieser Multiplikator muss allerdings insbesondere bei kleinen Sketchmengen sorgfältig ausgewählt werden, damit der Wertebereich in dichten Netzwerkregionen nicht nach oben hin überschritten wird.

Neben der komplexen Konfiguration hat das beschriebene Verfahren noch weitere Nachteile. Zum einen können nur sehr einfache Aggregierungsfunktionen wie COUNT und SUM direkt abgebildet werden, zum anderen kann die Genauigkeit der Schätzung auch mit deutlicher Vergrößerung der Anzahl der Sketches nicht signifikant unter 4 % gesenkt werden. Aus diesen Gründen wird in den folgenden Abschnitten eine alternative Datenstruktur vorgestellt und evaluiert, die speziell für weniger dichte Netzwerke eine flexiblere und genauere Aggregierung von Sensordaten ermöglicht.

### 4.3.1 Verlustfreie Datenübertragung

Die bisher beschriebene Verbreitungsstrategie kann auch ohne eine probabilistische Datenstruktur umgesetzt werden. Dafür sendet jeder Knoten regelmäßig eine Liste mit allen ihm bekannten Sensorwerten an seine Nachbarn. Jeder Knoten kombiniert dann im nächsten Schritt alle empfangenen Listen miteinander und sendet die so entstandene Liste wiederum an seine Nachbarn. Um eine derartige Kombination zu ermöglichen, müssen die Listeneinträge neben den eigentlichen Sensorwerten zusätzliche Informationen enthalten. Analog zum bisher untersuchten Verfahren kann jeder Listeneintrag mit einer verbleibenden Gültigkeitsdauer versehen werden. Dieser Gültigkeitswert wird beim Einfügen beziehungsweise Aktualisieren eines Sensorwerts mit dem Maximalwert initialisiert und bei jeder zukünftigen Übertragung um eins reduziert. Listeneinträge mit einer Gültigkeitsdauer von null werden aus der Liste entfernt und nicht weiter übertragen. Damit die Empfänger die Einträge der eingehenden Listen miteinander kombinieren können, müssen diese zusätzlich noch einen Identifikator enthalten, der angibt, von welchem Knoten die jeweiligen Sensordaten stammen.

Für die meisten Anwendungen sind Sensorwerte zwischen 0 und 255 ausreichend. Der Wertebereich der verbleibenden Gültigkeitsdauer hängt primär von der Größe des gewünschten Beobachtungsgebiets ab. Da bei dem hier beschriebenen Ansatz – im Unterschied zu den TTL-Zählern der Soft State Sketches – keine leeren Positionen übertragen werden, entfällt die Notwendigkeit, eine Gültigkeitsdauer von null zu codieren. Wenn wie in der vorherigen Simu-

lationsstudie jeder Knoten die Sensordaten seiner Nachbarn in einem Radius von acht Hops ausgewertet, dann werden dementsprechend auch nur acht Symbole pro TTL-Zähler benötigt. Generell gilt, dass mit  $N$  Bit bis zu  $2^N$  Hops gezählt werden können und somit der Speicherbedarf der TTL-Zähler vergleichsweise moderat ausfällt. Als Identifikator eines Listeneintrags kann die in WLANs ohnehin schon existierende eindeutige MAC-Adresse verwendet werden. Ausgehend von der beschriebenen Konfiguration ergibt sich pro Listeneintrag ein Speicherbedarf von  $3 + 8 + 48 = 59$  Bit.

Der Speicherbedarf von 64 Sketches mit entsprechenden TTL-Zählern beträgt unkomprimiert 4096 Bit. Dementsprechend benötigt diese Datenstruktur erst ab 70 Nachbarn im Beobachtungsgebiet weniger Speicherplatz als die verlustfreie Übertragung der beschriebenen Liste. Zwar befinden sich in dem betrachteten Netzwerk aus Mannheimer Access Points im Durchschnitt etwa 700 Nachbarn in einem Radius von 8 Hops, dennoch zeigt dieser Vergleich, dass es für weniger dichte Netzwerke oder bei wenig teilnehmenden Geräten sinnvoll sein kann, die Daten direkt zu übertragen.

Den größten Teil des Speicherbedarfs eines solchen Listeneintrags macht der Identifikator des Knotens aus. Die als Identifikator verwendete MAC-Adresse soll möglichst global eindeutig sein und benötigt deshalb mehr Speicherplatz, als in diesem Szenario unbedingt notwendig. Sei für die nachfolgende Überlegung  $A$  die Anzahl der Knoten im Netzwerk. Unter der Annahme, dass die Zusammensetzung und Anzahl der Knoten konstant sind, könnte eine zentrale Instanz jedem Knoten vor der Inbetriebnahme des Netzwerks eine eindeutige Zahl  $I_K \in \{1, \dots, A\}$  zuordnen. Alternativ könnte hierfür während der Laufzeit auch ein dezentrales Verfahren herangezogen werden. Aus der Sicht eines einzelnen Datenpakets sind allerdings auch diese Identifikatoren nur dann optimal codiert, wenn alle im Netzwerk vergebenen Identifikatoren in dem übertragenen Datenblock auftreten. Sind in dem Paket beispielsweise nur zwei Identifikatoren enthalten, so können diese effizienter mit eins und zwei nummeriert werden. Auch wenn die zuvor beschriebene Vergabe der Identifikatoren aus der Perspektive eines einzelnen Pakets nicht optimal erfolgt ist, kann dieser Ansatz dennoch den Speicherbedarf der ausgetauschten Listen in kleinen Szenarien deutlich reduzieren. Wenn beispielsweise  $A = 255$  ist, dann benötigt jeder Identifikator anstatt 6 Byte nur noch 1 Byte. Allerdings muss dann das zusätzliche Datenaufkommen für die Zuordnung der global gültigen Identifikatoren gegengerechnet werden. Außerdem müssen Sonderfälle wie beispielsweise der Beitritt eines neuen Knotens berücksichtigt werden. Aus diesen Gründen kann der Speicherplatz der Identifikatoren durch eine Reduktion des verwendeten Wertebereichs nur mit signifikantem Mehraufwand gesenkt werden.

Alternativ können die Identifikatoren mit einem entropiebasierten Kompressionsverfahren komprimiert werden. Dafür werden aber die relativen Häufigkeiten aller  $A$  Symbole benötigt, die zusätzlich übertragen werden müssen. Da die meisten Symbole nicht auftreten, ist es in dem oben genannten Beispiel ausreichend, die relativen Häufigkeiten der beiden enthaltenen Identifikatoren zu übertragen. Bei der Codierung der Information, welche der übertragenen Wahrscheinlichkeiten sich auf welchen Identifikator beziehen, entsteht allerdings letztlich die



gleiche Problematik wie zuvor. Dementsprechend kann der Speicherbedarf der ausgetauschten Listen durch eine derartige Kompression der Identifikatoren nicht reduziert werden.

### 4.3.2 Aufbau und Eigenschaften

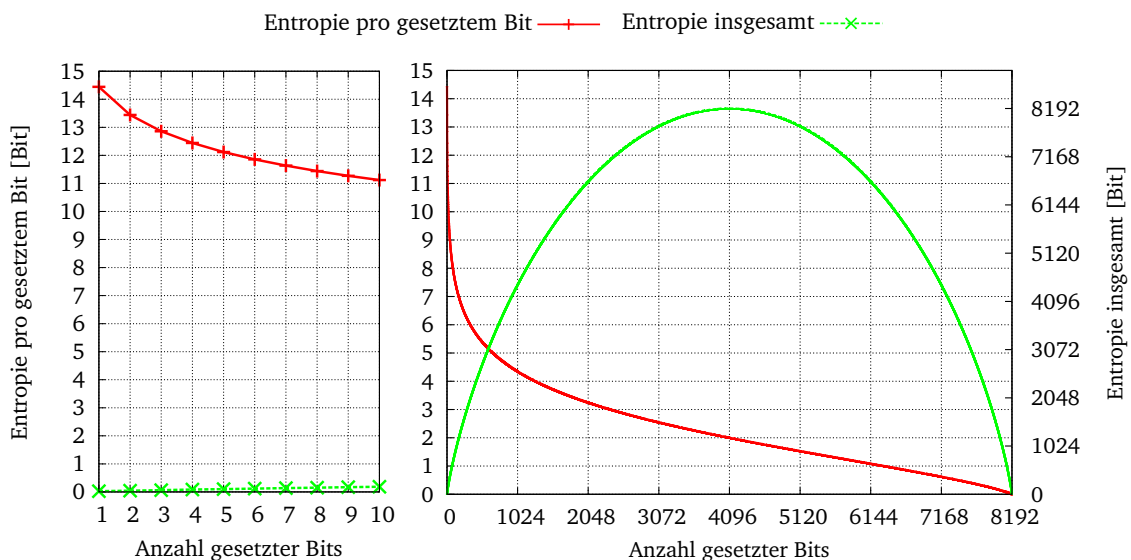
Der im Folgenden präsentierte Datenstruktur liegt die Idee zugrunde, die Zahlen  $I_K \in \{1, \dots, A\}$  als Positionen in einem Bitarray zu interpretieren. In diesem Bitarray ist das Bit  $S_i$  mit  $i = 0 \dots A-1$  daher nur dann gesetzt, wenn der entsprechende Identifikator  $I_K$  in dem übertragenen Paket enthalten ist. Diese Verwendung ähnelt einem *Bloomfilter* [24]. Bloomfilter können beispielsweise verwendet werden, um festzustellen, welche Daten in einem kontinuierlich fließenden Datenstrom bereits vorgekommen sind und welche erstmals auftreten. Dafür wird das Bit des Arrays inspiziert, welches dem Zahlenwert des aktuell betrachteten Datums entspricht. Falls dieses bisher noch nicht gesetzt wurde, ist das aktuell betrachtete Datum noch nicht aufgetreten. Dementsprechend wird in einem solchen Fall die aktuelle Beobachtung durch das Setzen des jeweiligen Bits registriert. Sollte im Umkehrschluss das Bit bereits gesetzt gewesen sein, kann davon ausgegangen werden, dass das entsprechende Datum bereits früher aufgetreten ist.

Im Unterschied zu den Bloomfiltern wird bei den hier präsentierten *Bloomfilter-Maps* (BF-Maps) [141] noch zusätzlich eine Liste mit den verbleibenden Gültigkeitswerten und den Sensordaten übertragen. Diese Liste ist nach dem jeweiligen Auftreten des zugehörigen Identifikators innerhalb des Bitarrays sortiert. Bei  $A = 8192$  benötigt das Bitarray beispielsweise immer 8192 Bit, unabhängig davon, wie viele Identifikatoren tatsächlich im Paket enthalten sind. Die direkte Übertragung eines einzelnen Identifikators benötigt jedoch nur 13 Bit, so dass mit dem gleichen Datenvolumen etwa 630 Identifikatoren übertragen werden können. Dementsprechend kann in diesem Beispiel erst ab 631 gleichzeitig übertragenen Identifikatoren durch die Verwendung eines Bitarrays Speicherplatz gespart werden.

Der eigentliche Vorteil von BF-Maps gegenüber der in Abschnitt 4.3.1 beschriebenen Variante liegt aber in dem kleinen Alphabet von nur zwei Symbolen. Ist die Anzahl der gesetzten Bits innerhalb des Arrays  $x$  bekannt, so können die beiden relativen Häufigkeiten  $p_0(x)$  und  $p_1(x)$  bestimmt werden. Folglich reicht die zusätzliche Übertragung der Anzahl der Einsen aus, damit das Bitarray entropieoptimal komprimiert ausgetauscht werden kann. Die Entropie eines Bitarrays kann in Abhängigkeit von der Anzahl der gesetzten Bits  $x_1$  und der Größe des Arrays  $A$  gemäß Formel 4.13 berechnet werden.

$$\begin{aligned}
 p_0(x) &= (A - x_1) / A \\
 p_1(x) &= x_1 / A \\
 e_0(x) &= -p_0(x) \cdot \log_2(p_0(x)) \\
 e_1(x) &= -p_1(x) \cdot \log_2(p_1(x)) \\
 e(x) &= A \cdot (e_0(x) + e_1(x))
 \end{aligned} \tag{4.13}$$

Abbildung 4.13 zeigt exemplarisch die Entropie eines Arrays mit 8192 Bit in Abhängigkeit von der Anzahl der gesetzten Bits. Falls beispielsweise in einer derartigen Datenstruktur nur ein Bit gesetzt ist, so ergibt sich bei  $A = 8192$  eine Entropie von nur 14,4 Bit. Dies sind nur 1,4 Bit mehr als der Speicherbedarf eines entsprechenden Identifikators. Schon bei drei Identifikatoren liegt die Entropie pro Identifikator mit 12,8 Bit unter dem Speicherbedarf der einzelnen Zahlen. Wie aus der Abbildung ersichtlich ist, verstärkt sich dieser Trend mit steigender Anzahl von eingefügten Elementen weiter. Beispielsweise besitzt diese Datenstruktur mit 1000 gesetzten Bits eine Entropie von etwa 4385 Bit, was circa 4,4 Bit pro Identifikator entspricht. Die Ursache für den Entropieunterschied zwischen der Zahlenliste und dem Bitarray liegt hauptsächlich in dem Umstand begründet, dass die Liste im Gegensatz zum Bitarray eine beliebige Reihenfolge und auch Mehrfachnennungen der Identifikatoren ermöglicht. Allerdings werden diese Optionen in dem angedachten Anwendungsfall nicht benötigt.

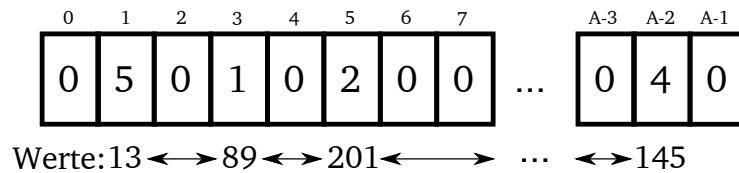


**Abbildung 4.13:** Entropie eines acht Kilobyte großen Bitarrays in Abhängigkeit von der Anzahl der gesetzten Bits.

Anstatt die verbleibenden Gültigkeitsdauern in der zusätzlich geführten Liste mit den Sensorwerten zu speichern, werden im Folgenden die TTL-Zähler direkt in dem Array hinterlegt. Auch wenn sich hierdurch die Anzahl der unterschiedlichen Symbole vergrößert, wird im Mittel aufgrund der zusätzlichen Kompression der TTL-Zähler insgesamt weniger Speicherplatz benötigt. Wie zuvor werden die Sensorwerte auch hier unkomprimiert in einer nach der Reihenfolge der zugehörigen Positionen sortierten Liste gespeichert.

Der beschriebene Aufbau der BF-Map wird in Abbildung 4.14 anhand eines Beispiels visualisiert. Vor der Übertragung der BF-Map an die Nachbarn wird das Array komprimiert. Damit das Array vom Empfänger wieder dekomprimiert werden kann, enthält das zu übertragene

Packet neben der Liste mit den Sensorwerten auch die jeweiligen Häufigkeiten der einzelnen TTL-Werte.



**Abbildung 4.14:** Exemplarischer Aufbau einer Bloomfilter-Map, bestehend aus einem TTL-Array und der zugehörigen Liste mit Sensorwerten.

In einem realen Einsatz ist die Anzahl der teilnehmenden Knoten  $A$  unbekannt und über die Zeit nicht konstant. Des Weiteren verfügen die Geräte nicht über einen effizienten Mechanismus, der dezentral feststellt, welche Geräte an der Aggregation teilnehmen wollen, und diesen dann eindeutige Nummern zuweist. Dementsprechend kann nicht davon ausgegangen werden, dass die teilnehmenden Knoten eindeutige Identifikatoren aus dem Wertebereich  $1, \dots, A$  besitzen. In den meisten Einsatzszenarien stehen selbst netzwerkweit eindeutige und optimal codierte Identifikatoren nicht zur Verfügung. Aus diesen Gründen wird im Folgenden auf die ohnehin vorhandenen und mit hinreichender Wahrscheinlichkeit weltweit eindeutigen MAC-Adressen der Geräte zurückgegriffen. Da allerdings der Wertebereich von  $0, \dots, 2^{48}$  viel zu groß für ein entsprechendes TTL-Array ist und dieses im Vergleich zu seiner Größe nur äußerst gering gefüllt wäre, werden anstatt der MAC-Adressen deren Hashwerte herangezogen, die mittels Modulo auf ein TTL-Array der gewünschten Größe  $B$  verteilt werden. Die gleiche Abbildungsstrategie wird sowohl bei der Verteilung der einzufügenden Elemente über die einzelnen Sketches als auch bei den weiter oben erwähnten Bloomfiltern verwendet.

Da hierbei mehrere MAC-Adressen auf der gleichen Position innerhalb des Arrays abgebildet werden können, handelt es sich bei einer derartig verwendeten BF-Map um einen probabilistischen Ansatz. Im Unterschied zu den zuvor untersuchten Sketches sind bei dieser Strategie nur die Identifikatoren – nicht aber die eigentlichen Sensorwerte – von den potenziellen Überschneidungen betroffen. Im Ergebnis bedeutet dies, dass bereits enthaltene Werte von anderen Knoten, deren Identifikator auf die gleiche Position innerhalb des Arrays abgebildet werden, versehentlich überschrieben werden können. Ebenfalls können Sensorwerte einzelner Knoten durch die Kombination von mehreren empfangenen Listen verloren gehen. In der verwendeten Implementierung der Listenkombination werden immer die Einträge mit der höheren verbleibenden Gültigkeitsdauer bevorzugt. Dementsprechend werden bei einer Überschreibung immer die Sensorwerte derjenigen Knoten überschrieben, die am weitesten von dem aktuellen Knoten entfernt sind. Dadurch wird zwar gewissermaßen das Beobachtungsgebiet teilweise verkleinert, allerdings entsprechen in jedem Fall alle enthaltenen Sensorwerte und deren Gültigkeitsdauern den real eingefügten Werten.

### 4.3.3 Fehlerabschätzung

Im Folgenden wird davon ausgegangen, dass die anhand der Identifikatoren generierten Hashwerte näherungsweise gleich verteilt sind. In diesem Fall beträgt die Wahrscheinlichkeit, mit der ein Identifikator auf eine bestimmte Position des aus  $B$  Zählern bestehenden Arrays abgebildet wird,  $1/B$ . Dementsprechend beträgt die Wahrscheinlichkeit, dass nach  $r$  Einfügeoperationen mindestens einmal ein Identifikator auf eine bestimmte Position abgebildet wurde,  $1 - (1 - 1/B)^r$ . Auf dieser Beobachtung aufbauend kann die Anzahl der im Durchschnitt gesetzten Positionen  $a$  gemäß Formel 4.14 abgeschätzt werden.

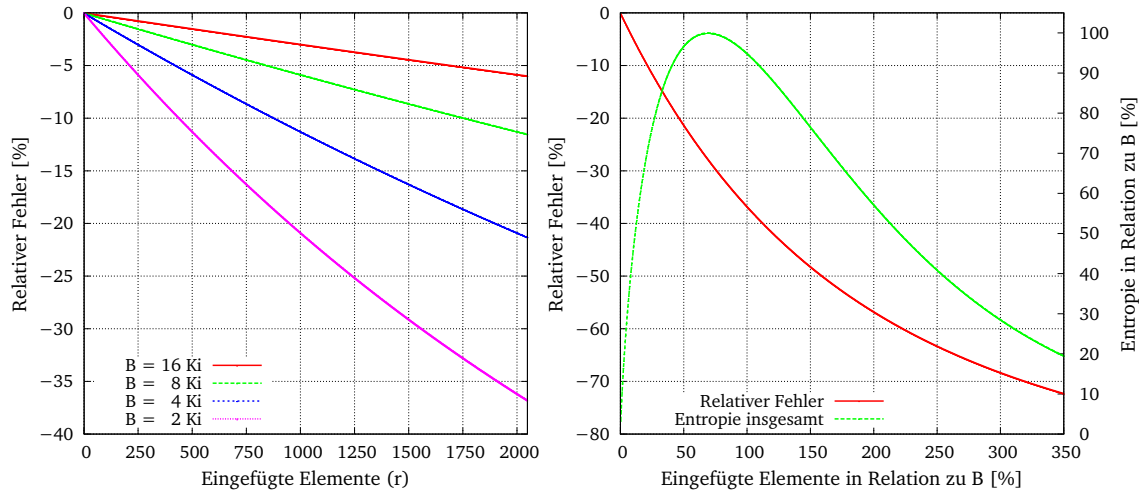
$$a(r, B) = B \cdot \left( 1 - \left( 1 - \frac{1}{B} \right)^r \right) \quad (4.14)$$

Da die Sensordaten unverändert in einer Liste vorliegen, kann mittels einer BF-Map prinzipiell jede Art von Aggregation berechnet werden. Welche Genauigkeit dabei zu erwarten ist, hängt neben den potenziellen Überschreibungen der Identifikatoren auch von der Werteverteilung der Sensordaten und den konkreten Berechnungsschritten ab. Um die Anzahl von Knoten mit einem bestimmten Merkmal zu bestimmen (COUNT), ist aber nur die Anzahl der gesetzten Zähler im Array von Bedeutung, sodass der zu erwartende durchschnittliche Fehler  $F_r$  mittels Formel 4.14 berechnet werden kann.

$$F_r = \frac{a(r, B)}{r} - 1 \quad (r \neq 0) \quad (4.15)$$

Auf der linken Seite von Abbildung 4.15 ist der im Durchschnitt zu erwartende relative Fehler für unterschiedlich große Arrays dargestellt. Im Unterschied zu den bisherigen Ansätzen kann bei dem auf diese Weise bestimmten COUNT der eigentliche Wert immer nur unterschätzt, aber nie überschätzt werden, sodass in dieser Abbildung ausnahmsweise der relative Fehler mit Vorzeichen angegeben ist. Da die Kollisionswahrscheinlichkeit mit der Anzahl der eingefügten Elemente steigt, steigt gleichzeitig auch der zu erwartende relative Fehler. Dementsprechend hängt der relative Fehler ausschließlich von dem Verhältnis zwischen der Anzahl der bereits eingefügten Elemente und der Größe des Arrays ab. Dieser Zusammenhang ist auf der rechten Seite der Abbildung dargestellt. Aufgrund der Kollisionen sind auch nach dem Einfügen von deutlich mehr als  $B$  Elementen noch Positionen unbelegt, sodass weiterhin eine Abschätzung vorgenommen werden kann. Aus diesem Grund wird auf der x-Achse der rechten Abbildung ein deutlich größerer Wertebereich dargestellt.

Nachdem in ein Array mit  $B = 1024$  Zählern 52 Elemente eingefügt wurden, beträgt der zu erwartende Fehler beispielsweise etwa 2,5 %. Werden hingegen 512 Elemente eingefügt, so steigt der relative Fehler auf circa 21 %. Dementsprechend muss bei diesem Ansatz die Größe des Arrays in Abhängigkeit von der zu erwartenden Anzahl von teilnehmenden Knoten im Beobachtungsgebiet und dem gewünschten Fehler gewählt werden. Gleichzeitig beeinflusst das Verhältnis zwischen den eingefügten Elementen und der Größe des Arrays auch dessen Entropie. Der rechten Abbildung kann entnommen werden, dass in dem genannten Beispiel



**Abbildung 4.15:** Relativer Fehler der COUNT-Abschätzung einer Bloomfilter-Map in Abhängigkeit von der Anzahl der eingefügten Elemente.

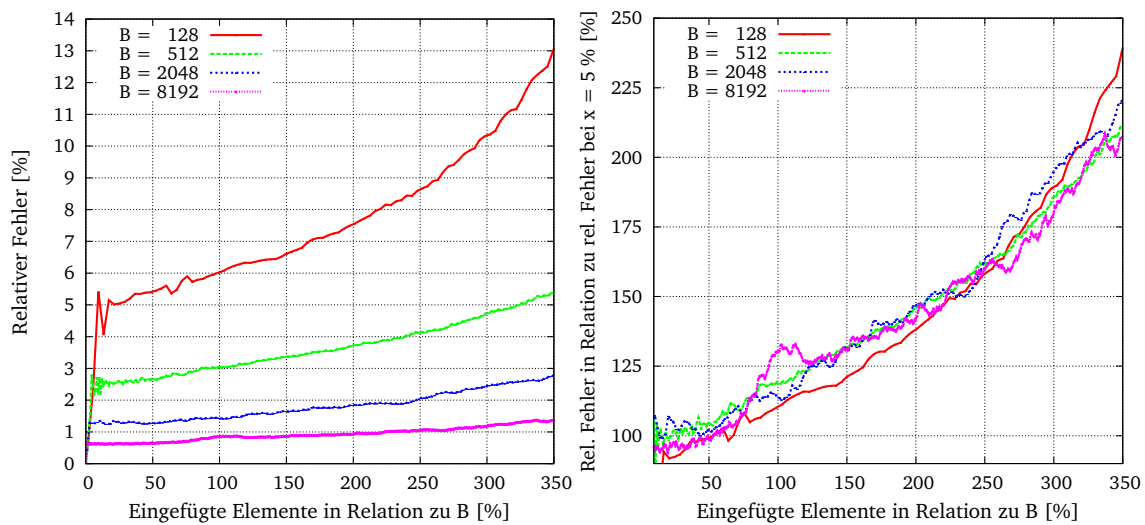
das Bitarray mit 52 Elementen optimal entropiecodiert einen Speicherplatz von circa 36 Byte benötigt, wohingegen das gleiche Array mit 512 Elementen etwa 124 Byte benötigt. Da sich die Wahrscheinlichkeiten verschieben, würde im letztgenannten Fall eine Verdopplung der Array-Größe den zu erwartenden Fehler fast halbieren, die Entropie aber nur um rund 63 % erhöhen.

Bisher wurde die Anzahl der eingefügten Elemente anhand der Anzahl der gesetzten Positionen innerhalb des Arrays abgeschätzt. Die durchschnittlich zu erwartende Anzahl der gesetzten Positionen  $a$  kann gemäß Formel 4.14 bestimmt werden und hängt ausschließlich von der Anzahl der eingefügten Elemente  $r$  und der Größe des Arrays  $B$  ab. Folglich kann nach einer Umstellung dieser Formel anstatt  $a$  direkt das im Durchschnitt zugehörige  $r$  als Schätzwert herangezogen werden. Die entsprechende Abschätzung ist in Formel 4.16 angegeben. Welcher Fehler dabei zu erwarten ist, kann allerdings nicht ohne ein statistisches Modell der Wertestreuung berechnet werden. Da die Datenstruktur ohnehin mit Hilfe einer Simulation untersucht werden soll, wird im Folgenden auf ein statistisches Modell verzichtet und der relative Fehler empirisch bestimmt.

$$r(a, B) = \frac{\log(1 - \frac{a}{B})}{\log(1 - \frac{1}{B})} \quad (4.16)$$

Abbildung 4.16 zeigt den empirisch bestimmten relativen Fehler für unterschiedlich große Arrays. Wie in den vorherigen Abbildungen wurde die Anzahl der eingefügten Elemente in Abhängigkeit von der Größe des jeweiligen Arrays eingezeichnet. Allerdings unterscheiden sich die Ergebnisse gemäß Formel 4.16 trotz der relativen Angabe deutlich voneinander und sind somit von der Größe des Arrays abhängig. Dies begründet sich vor allem daraus, dass die Formel auf dem durchschnittlichen Zusammenhang zwischen  $a$  und  $r$  aufbaut. Falls die Elemente hinreichend gleich über den Wertebereich verteilt sind, gleichen sich die im realen

Einsatz auftretenden Schwankungen des Zusammenhangs zwischen  $a$  und  $r$  im Mittel aus. Allerdings können sich während der Simulation diese Ungenauigkeiten nicht vollständig aufheben, weil dort der Betrag des relativen Fehlers ausgewertet wird. Dementsprechend ist der relative Fehler umso kleiner, je näher der konkret im Array vorliegende Zusammenhang dem im Durchschnitt zu erwartenden entspricht. Des Weiteren gilt, dass, je größer ein Array ist, desto mehr Elemente eingefügt werden müssen, um einen bestimmten Prozentsatz an gesetzten Positionen zu erreichen. Wenn die in einem Array vorliegenden Informationen als Stichprobe aus der Grundgesamtheit interpretiert werden, so ist deren Größe direkt von derjenigen des Arrays abhängig. Daher sinkt bei gleich bleibender Füllquote die zu erwartende Streuung mit der Größe des Arrays, sodass der Fehler der Abschätzung im Durchschnitt kleiner wird.



**Abbildung 4.16:** Relativer Fehler der verbesserten COUNT-Abschätzung einer Bloomfilter-Map in Abhängigkeit von der Anzahl der eingefügten Elemente.

Bei einem fast leeren Array ist die Wahrscheinlichkeit, dass ein einzufügendes Element auf einer bereits benutzten Position abgebildet wird, sehr gering, sodass die Anzahl der eingefügten Elemente mit einer hohen Genauigkeit geschätzt werden kann. Nachdem etwa  $0,05 \cdot B$  Elemente eingefügt wurden, erreicht der jeweilige relative Fehler das von der Größe des Arrays abhängige Grundniveau. Bei der Simulation werden die einzelnen Ergebnisse innerhalb eines bestimmten Intervalls gruppiert und ausgewertet. Da allerdings die Größe der verwendeten Intervalle für alle Konfigurationen identisch ist, sind die in der Abbildung 4.16 dargestellten Ergebnisse unterschiedlich genau aufgelöst, sodass die beschriebene Anfangsphase nicht immer gleich gut zu sehen ist. Auf der rechten Seite der Abbildung ist der relative Fehler unterschiedlicher Array-Größen in Relation zu dem zuvor beschriebenen Grundniveau eingezeichnet. Wie aus der Grafik ersichtlich ist, verläuft der relative Fehler in Bezug auf das jeweilige Grundniveau sowie das Verhältnis zwischen eingefügten Elementen und  $B$  für unterschiedlich große Arrays in etwa gleich. Da die Entropie von der Genauigkeit der Schätzung unabhängig ist,

kann für die Auswahl einer geeigneten Konfiguration auf die entsprechenden Ergebnisse in Abbildung 4.15 zurückgegriffen werden.

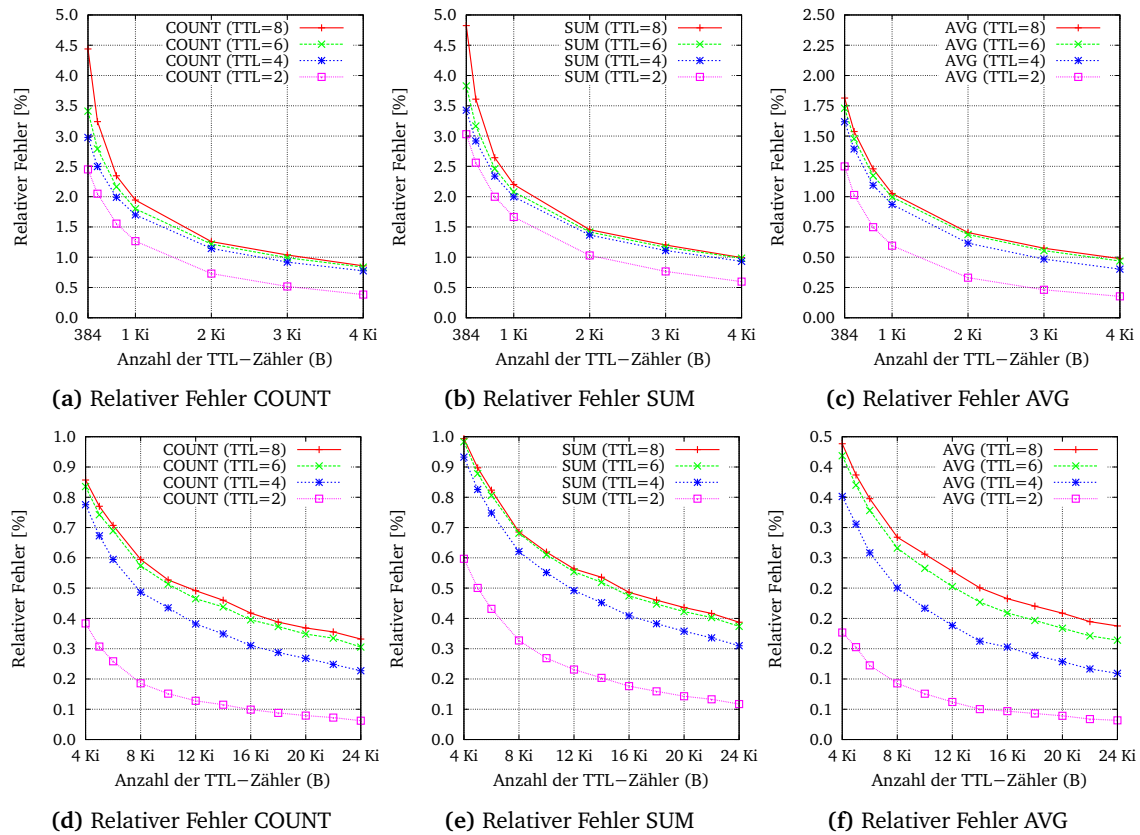
### 4.3.4 Simulationsergebnisse

Der relative Fehler für eine Abschätzung von COUNT hängt nicht nur von der Größe des Arrays, sondern auch von der Anzahl der eingefügten Elemente ab. Aus diesem Grund können die bisherigen Ergebnisse nicht direkt auf eine reale Anwendung übertragen werden. Zusätzlich wird für die Bestimmung von SUM und AVG die gesamte Datenstruktur inklusive der zusätzlich geführten Liste mit Sensorwerten benötigt. Deshalb wird im Folgenden erneut das Mannheimer Access-Point-Netzwerk aus Abschnitt 5.4 herangezogen.

In jedem Simulationsdurchlauf generiert jeder Knoten einen gleich verteilten Zufallswert zwischen 0 und 255 und fügt diesen mit der maximalen verbleibenden Gültigkeitsdauer in die eigene BF-Map ein. Anschließend werden die komprimierten BF-Maps zwischen den Knoten ausgetauscht und kombiniert, wobei bei jeder Übertragung alle TTL-Zähler um eins reduziert werden. Wie bei der Aggregation mittels Soft State Sketches, die in Abbildung 4.9 dargestellt ist, werden die lokalen Sensorwerte nach jeder Übertragung erneut in die lokale BF-Map eingefügt, um die entsprechenden TTL-Zähler zu erneuern. Wenn das Netzwerk eingeschwungen ist, bestimmt jeder Knoten die Anzahl seiner Nachbarn, die Summe der eingefügten Sensordaten und deren Durchschnitt. Die Anzahl der Nachbarn wird dabei gemäß Formel 4.16 bestimmt. Die Sensorwerte derjenigen Knoten, deren Identifikatoren aufgrund von Abbildungskollisionen überschrieben wurden, sind nicht in der zusätzlichen Liste enthalten und können dementsprechend auch nicht bei der Auswertung berücksichtigt werden. Um die Summe besser abschätzen zu können, wird aber zumindest die Summe der vorhandenen Sensorwerte mit Hilfe des Verhältnisses zwischen  $r$  und  $a$  extrapoliert. Im Unterschied dazu beruht die Berechnung des Durchschnitts ausschließlich auf den vorhandenen Sensorwerten. Nach der Berechnung vergleicht der Simulator für jeden Knoten die mit Hilfe der BF-Maps bestimmten Ergebnisse mit den realen Werten und berechnet die resultierenden relativen Fehler. Dieser Ablauf entspricht weitgehend der in Abschnitt 4.2.4 durchgeführten Simulationsstudie, die jeweiligen Ergebnisse direkt miteinander vergleichbar sind. Alle im Folgenden präsentierten Ergebnisse basieren jeweils auf 150 Simulationsdurchläufen.

In der vorherigen Simulation der Soft State Sketches wird ab einem Untersuchungsbereich von drei Hops die erforderliche Mindestanzahl von eingefügten Elementen so gut wie immer erreicht, sodass der durchschnittliche relative Fehler ab diesem Radius ausschließlich von der Anzahl der verwendeten Sketches abhängt. Im Unterschied dazu steigt bei einer BF-Map mit der Anzahl der teilnehmenden Knoten die Wahrscheinlichkeit einer Kollision der Identifikatoren, wodurch der relative Fehler steigt. Aus diesem Grund wurden in Abbildung 4.17 die Ergebnisse sowohl in Abhängigkeit von der Anzahl der TTL-Zähler als auch für unterschiedlich große Beobachtungsradien eingezeichnet. Generell kann bei allen drei Aggregierungsfunktionen festgestellt werden, dass eine Vergrößerung des Beobachtungsgebiets bei einer gleich bleibenden Array-Größe immer zu einer Verschlechterung der Ergebnisse führt. Da sich mit stei-



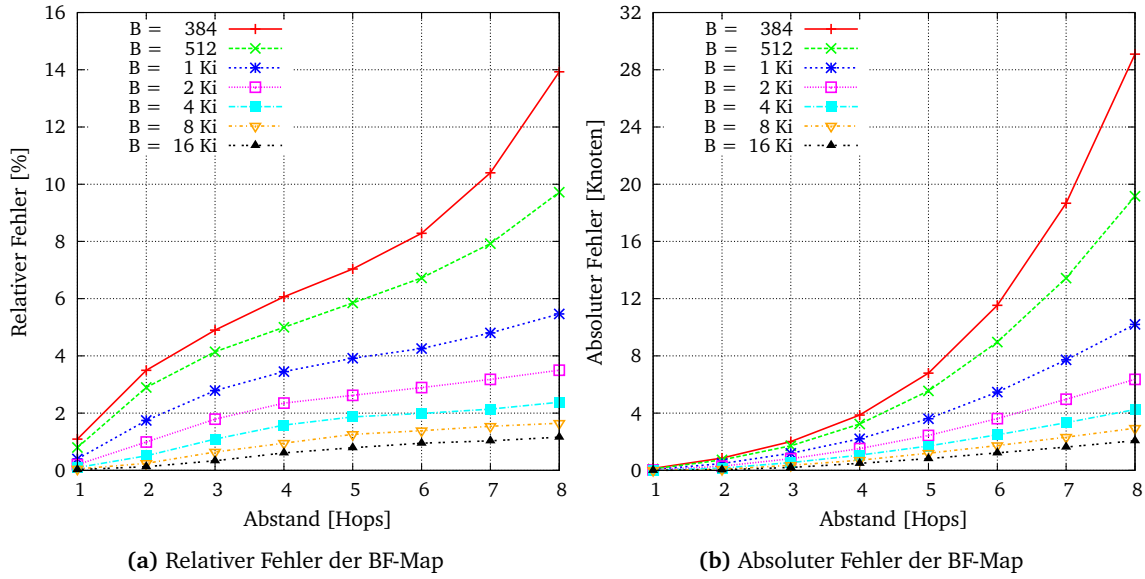


**Abbildung 4.17:** Relativer Fehler einer Bloomfilter-Map für unterschiedliche Aggregierungsfunktionen in Abhängigkeit von der Anzahl der TTL-Zähler und der Größe des Beobachtungsgebiets.

gender Knotenanzahl auch die Wahrscheinlichkeit von Abbildungskollisionen vergrößert, ist diese Entwicklung für COUNT und SUM naheliegend. Die Kollisionen betreffen zwar auch die Berechnung von AVG, allerdings vergrößert sich gleichzeitig die Größe der Stichprobe, anhand derer der eigentliche Durchschnitt abgeschätzt wird. In der vorliegenden Konfiguration kann dies die Verschlechterung aufgrund der Kollisionen aber nicht vollständig ausgleichen. Da die Summe der Sensorwerte extrapoliert wird, liegen die Ergebnisse von SUM immer geringfügig über denen von COUNT. Beispielsweise kann COUNT mit  $B = 1024$  mit einem relativen Fehler von circa 1,9 % bestimmt werden, wohingegen der vergleichbare Fehler von SUM etwa 2,2 % beträgt. Die höchste Genauigkeit kann bei der Bestimmung von AVG erzielt werden. Bei der zuvor genannten Konfiguration kann der Durchschnitt mit einem relativen Fehler von rund 1,0 % berechnet werden. Diese hohen Genauigkeiten können als Indiz dafür gewertet werden, dass auch andere Aggregierungen, die anhand einer Stichprobe auf die Grundgesamtheit schließen, ein ähnlich niedriges Fehlerniveau erreichen können.

Wie auch bei den Soft State Sketches können BF-Maps nicht nur für die Auswertung aller Knoten innerhalb des jeweiligen Beobachtungsgebiets verwendet werden, sondern auch, um

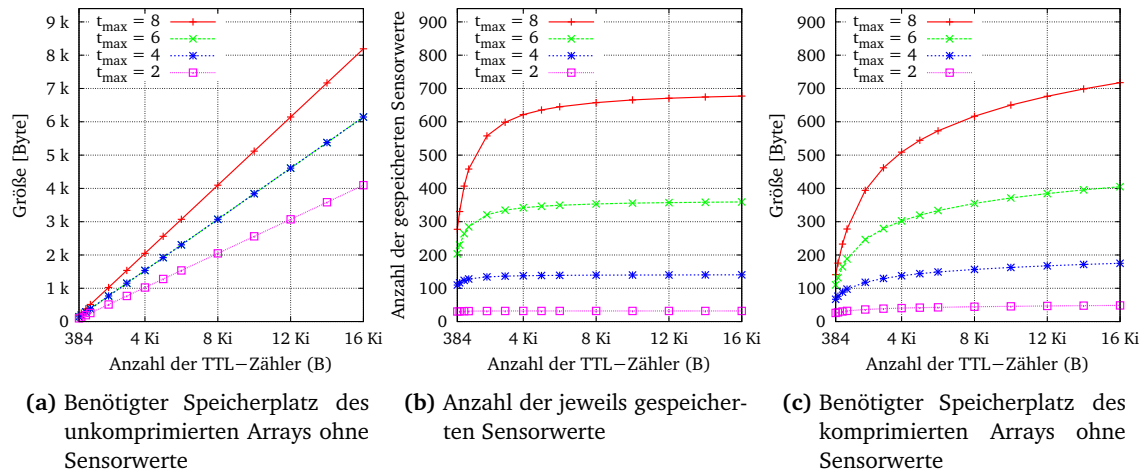




**Abbildung 4.18:** Relativer Fehler einer Bloomfilter-Map bezüglich der Abschätzung der Anzahl der Nachbarn in einem bestimmten Abstand.

zwischen unterschiedlich weit entfernten Knoten zu differenzieren. Beispielsweise kann auf diese Weise der Durchschnitt aller Knoten innerhalb von zwei Hops mit den Sensorwerten von weiter entfernten Knoten verglichen werden, um so räumliche Veränderungen der Sensorwerte abschätzen zu können. Für den direkten Vergleich mit den vorherigen Simulationen zeigt Abbildung 4.18 den relativen und den absoluten Fehler der Abschätzung der Anzahl der Nachbarn in einem bestimmten Abstand. Hierbei werden die benötigten Werte analog zu den Vergleichswerten schrittweise verrechnet. Beispielsweise kann die Anzahl aller Knoten in einem Radius von drei Hops bestimmt werden, indem nur diejenigen TTL-Zähler berücksichtigt werden, deren verbleibende Gültigkeitsdauer mehr als  $t_{max} - 3$  beträgt. Um nun die Anzahl der Knoten im Abstand von exakt drei Hops zu ermitteln, wird die Anzahl der Knoten innerhalb eines Radius von zwei Hops von der Anzahl der Knoten innerhalb eines Radius von drei Hops subtrahiert. Dieser Berechnungsprozess führt zusammen mit der Tatsache, dass die Identifikatoren weiter entfernter Knoten von näher liegenden Knoten überschrieben werden können, zu einem Schätzfehler, der analog zum Abstand steigt. Wenn beispielsweise ein 1024 Zähler großes Array verwendet wird, beträgt der relative Fehler bei der Bestimmung der Anzahl der Nachbarn in einem Abstand von drei Hops circa 2,9 %. Zwar steigt der Fehler bei einem Abstand von acht Hops auf etwa 5,5 %, allerdings dürfte auch dieses Ergebnis für zahlreiche Anwendungen ausreichend sein.

Insgesamt werden bei der Aggregation mit BF-Maps vergleichsweise niedrige Fehlerwerte erreicht. Dieser Umstand hängt auch von dem jeweils dargestellten Wertebereich der Konfigurationsparameter ab. Da sich aber im Durchschnitt etwa 700 Nachbarn in einem Radius von 8 Hops befinden, ist die Verwendung eines Arrays mit weniger als 384 Zählern aufgrund der



**Abbildung 4.19:** Von Bloomfilter-Maps benötigter Speicherplatz in Abhängigkeit von der Anzahl der TTL-Zähler (B) sowie der maximal verbleibenden Gültigkeitsdauer.

Abbildungskollisionen nicht sinnvoll. Um die Einsatzmöglichkeiten der unterschiedlichen Konfigurationen abzuschätzen, ist auf der linken Seite von Abbildung 4.19 der Speicherbedarf des Arrays für unterschiedlich große Beobachtungsgebiete dargestellt. In der Mitte der Abbildung ist die Anzahl der jeweils gespeicherten Sensorwerte zu sehen. Der insgesamt benötigte Speicherplatz einer BF-Map ergibt sich durch die Addition der Größe des Arrays und der Größe der entsprechenden Liste mit den Sensorwerten. Wie zuvor bereits angesprochen, kann das Array einer BF-Map entropieoptimal komprimiert werden. Für die Dekompression muss zusätzlich zur eigentlichen Datenstruktur die Anzahl der Einträge mit den jeweiligen Gültigkeitsdauern  $1 \dots t_{max}$  übertragen werden. Der durchschnittlich benötigte Speicherplatz des komprimierten Arrays inklusive der zusätzlichen Zähler ist auf der rechten Seite der Abbildung zu sehen. Da die Sensorwerte in dem untersuchten Szenario nicht komprimiert werden, ergibt sich das benötigte Datenvolumen einer komprimierten Übertragung aus der Größe des komprimierten Arrays und der Größe der Liste mit den Sensorwerten.

Bei dem weiter oben aufgegriffenen Beispiel von  $B = 1024$  benötigt das Array bei dem größten untersuchten Beobachtungsgebiet 512 Byte. In der zugehörigen Liste werden im Durchschnitt 458 Sensorwerte gespeichert. Dementsprechend werden insgesamt 970 Byte benötigt. Durch die beschriebene Kompression kann der Speicherbedarf des Arrays im Durchschnitt auf 262 Byte reduziert werden, was 4,6 Bit pro enthaltenem Identifikator entspricht. Mit den zusätzlichen 16 Byte für die Zähler und den Sensorwerten ergibt sich ein Speicherbedarf von 736 Byte. In diesem Fall macht die Liste mit den Sensordaten eindeutig den größten Teil des benötigten Speicherplatzes aus. Um den Speicherbedarf zu senken, kann beispielsweise die Größe des Arrays auf 384 gesetzt werden. Durch Kompression kann der Speicherbedarf eines Arrays dieser Größe von 192 Byte auf 124 Byte gesenkt werden. Aufgrund der höheren Kollisionswahrscheinlichkeit reduziert sich die Anzahl der Sensorwerte in der Liste auf etwa

276. Inklusive der zusätzlichen TTL-Zähler benötigt die komprimierte Datenstruktur 416 Byte. Der relative Fehler von COUNT und SUM liegen bei dieser Konfiguration deutlich unter 5 %. Obwohl durchschnittlich nur 276 von 700 Sensorwerten in der Liste enthalten sind, liegt der relative Fehler bei der Berechnung des Durchschnitts nur bei etwa 1,8 %.

Die kompakteste Variante der untersuchten BF-Maps besteht aus einem Array mit 384 Zählern und benötigt insgesamt 416 Byte. Für eine vergleichbare Genauigkeit bei der Abschätzung von COUNT und SUM werden mindestens 192 Sketches benötigt, welche komprimiert circa 510 Byte Speicherplatz belegen. Sollen sowohl COUNT als auch SUM bestimmt werden, müssen sogar zwei derartige Sketchmengen verwendet werden. Die BF-Map erreicht somit eine deutlich höhere Genauigkeit als die Sketches und benötigt dabei signifikant weniger Speicherplatz. Dieser Effekt verstärkt sich sogar, wenn das Beobachtungsgebiet verkleinert wird und damit die Anzahl der berücksichtigten Elemente pro Datenstruktur sinkt. Wird beispielsweise der Radius von den bisher besprochenen 8 Hops auf 4 Hops reduziert, benötigt die komprimierte BF-Map mit 384 Zählern im Durchschnitt nur rund 60 Byte. Inklusive der zusätzlichen TTL-Zähler und den Sensorwerten werden 178 Byte benötigt. Aufgrund der reduzierten Anzahl von berücksichtigten Elementen verringert sich der relative Fehler auf 2,9 % für COUNT und 3,4 % für SUM. Während PCSA diese Genauigkeit selbst mit 512 Sketches nicht ganz erreicht, kann als Referenz PCSA-LT mit 256 Sketches herangezogen werden, die bei einem Radius von 4 Hops komprimiert circa 443 Byte benötigen. In diesem Beispiel belegen die BF-Maps bei vergleichbarer Genauigkeit nur etwa 40 % des Speicherplatzes von PCSA-LT. Da hierbei auch die eigentlichen Sensorwerte erhalten bleiben, wird auch bei SUM, AVG oder einer gar nicht mit PCSA darstellbaren Aggregierungsfunktion kein zusätzlicher Speicherplatz benötigt. Ein Vorteil von PCSA ist allerdings die Möglichkeit, den Speicherplatz auf Kosten der Genauigkeit drastisch zu reduzieren. Beispielsweise erreichen zwei PCSA-Datenstrukturen mit jeweils 16 Sketches und einem Speicherbedarf von insgesamt circa 100 Byte zwar nur eine Genauigkeit von rund 16 %, aber eine BF-Map mit vergleichbarer Größe ist – zumindest in einem dichten Netzwerk – überhaupt nicht möglich.

## 4.4 Bloomfilter-Arrays

In einer BF-Map werden die eigentlichen Sensorwerte in einer separaten Liste gespeichert. Einerseits wird hierdurch eine hohe Genauigkeit erreicht, andererseits steigt der Speicherbedarf linear zur Knotenanzahl. Dieser Umstand wird durch die Abbildungskollisionen zwar eingedämmt, dennoch ist der Speicherbedarf stark von der Anzahl der Knoten im Beobachtungsgebiet abhängig. Im Vergleich hierzu sind die Sketches darauf ausgelegt, mit  $\mathcal{O}(\log n)$  Bit bis zu einer Kardinalität von  $n$  zählen zu können. Die Weiterentwicklungen, welche allerdings nicht mit dem Soft-State-Ansatz verbunden werden können, reduzieren den benötigten Speicherplatz sogar auf  $\mathcal{O}(\log \log n)$ . Um die Lücke zwischen FM-Sketches und BF-Maps zu schließen, wird in diesem Abschnitt ein neues Verfahren vorgestellt, das bei der dezentralen Aggregierung

eine höhere Genauigkeit als die FM-Sketches erreicht, aber ebenfalls nur  $\mathcal{O}(\log n)$  Speicherplatz benötigt.

### 4.4.1 Datenstruktur und Schätzfunktion

Grundsätzlich kann der lineare Zusammenhang zwischen dem Speicherbedarf und der Anzahl der eingefügten Elemente nur entkoppelt werden, wenn die Elemente innerhalb der Datenstruktur ungleichmäßig verteilt werden und sich hierbei gegenseitig überschreiben können. In einem einzelnen FM-Sketch wird dafür eine geometrische Verteilung verwendet. Diese Strategie kann auch auf die im vorherigen Abschnitt untersuchte Abwandlung des Bloomfilters übertragen werden. Anstatt eines einzelnen Bloomfilters mit  $B$  Positionen werden in dieser Konstellation  $L$  Bloomfilter der gleichen Größe verwendet, die in diesem Abschnitt mit  $A_0 \cdots A_{L-1}$  bezeichnet werden. Da bei der zuvor präsentierten BF-Map der Bloomfilter auch schon bei der Namensgebung Pate stand, wird die in diesem Abschnitt vorgestellte Datenstruktur im Folgenden *Bloomfilter-Array* (BF-Array) genannt.

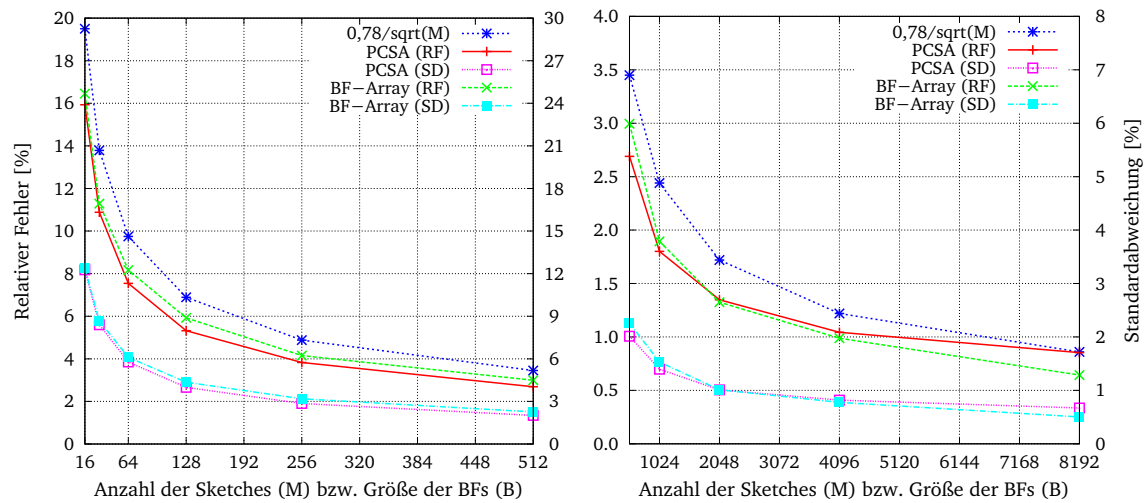
Soll ein Element  $x$  in ein BF-Array eingefügt werden, wird der Bloomfilter  $A_i$  mit  $i = l_1(\text{hash}(x)/B)$  ausgewählt. Innerhalb dieses wird das Element dann auf die Position  $j = \text{hash}(x) \bmod B$  abgebildet. Auch wenn die Daten auf eine andere Weise interpretiert werden, entstehen aufgrund der identischen Einfügeprozedur die gleichen Bitmuster wie bei einer Menge von  $L$  Sketches mit jeweils  $M = B$  Positionen. Im Unterschied zu PCSA werden die Rohdaten aber nicht zeilenweise, sondern spaltenweise interpretiert. Beispielsweise entspricht die erste Spalte einer PCSA-Datenstruktur, also die jeweils erste Position aller Sketches, dem ersten Bloomfilter eines BF-Arrays. Die Wahrscheinlichkeit, dass ein Element in  $A_i$  abgebildet wird, beträgt  $2^{-(i+1)}$ . Da somit alle Abbildungen innerhalb eines bestimmten Bloomfilters immer auf der gleichen Auswahlwahrscheinlichkeit basieren, kann mit Hilfe von Formel 4.16 die Anzahl der eingefügten Elemente abgeschätzt werden. Durch die Addition dieser Einzelergebnisse kann wiederum die Anzahl der insgesamt in die Datenstruktur eingefügten Elemente abgeschätzt werden.

Die Ergebnisse aus Abschnitt 4.3 zeigen, dass die Genauigkeit der Einzelschätzungen sowohl von  $B$  als auch vom Verhältnis zwischen der Anzahl der eingefügten Elemente und der Größe des Bloomfilters abhängen. Solange aber in jeden Bloomfilter maximal circa  $2-3 \cdot B$  Elemente eingefügt wurden, gleichen sich die Fehlerschwankungen zwischen den unterschiedlich stark gefüllten Bloomfilter in etwa aus. Aufgrund der geometrischen Verteilung werden allerdings einige der Bloomfilter über dieses Maß hinaus gefüllt und sollten nicht mehr für die Schätzung verwendet werden. Dies gilt insbesondere dann, wenn alle Positionen verwendet werden und aus diesem Grund überhaupt keine sinnvolle Schätzung mehr vorgenommen werden kann. Daher fließen nur die Ergebnisse derjenigen Bloomfilter in die Schätzung ein, in denen weniger als 80 % der Positionen belegt sind. Durch die geometrische Verteilung ist sichergestellt, dass bei einem sequenziellen Durchlauf der Spalten der Datenstruktur nach der ersten Spalte, die diese Bedingung erfüllt, nur noch Bloomfilter kommen, die ebenfalls weniger als 80 % belegte Positionen aufweisen. Insofern nicht bereits  $A_0$  die Bedingung erfüllt, teilt diese Su-

che das BF-Array in genau zwei Bereiche auf. Der erste Bereich besteht aus den Bloomfilter  $A_0 \dots A_i$  mit  $i \in \{1, \dots, L-1\}$ , die aufgrund einer zu hohen Anzahl von verwendeten Positionen nicht bei der Auswertung berücksichtigt werden. Im zweiten Bereich befinden sich die verbleibenden Bloomfilter  $A_{i+1} \dots A_L$ . Falls bereits  $A_0$  die Bedingung erfüllt, ist es ausreichend, alle Schätzungen einfach zu addieren. Sollte dies nicht der Fall sein, werden bei einer Addition der Schätzungen des zweiten Bereichs nicht mehr alle Elemente berücksichtigt, sodass das Ergebnis extrapoliert werden muss. Hierfür wird die Summe der Schätzungen mit  $2^i$  multipliziert.

#### 4.4.2 Simulationsergebnisse

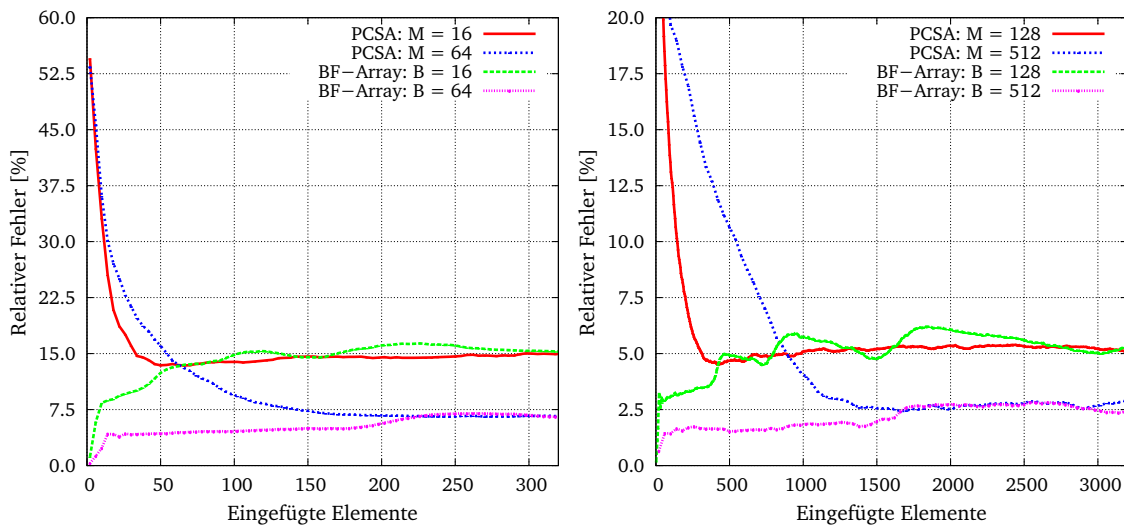
In der Simulation wird die Genauigkeit der beschriebenen Abschätzung mit PCSA verglichen. Dafür wird wie in Abschnitt 4.1.2 eine leere Datenstruktur erzeugt, in welche schrittweise zufällige Elemente eingefügt werden. Nachdem bereits  $40 \cdot L$  Elemente eingefügt wurden, wird nach jedem weiteren eingefügten Element die jeweilige Abschätzung mit dem realen Wert verglichen, und der daraus resultierende relative Fehler wird bestimmt. Diese Auswertung wird so lange fortgesetzt, bis  $4000 \cdot L$  Elemente eingefügt wurden. Alle präsentierten Werte basieren auf dem Durchschnitt von hundert derartigen Simulationsdurchläufen.



**Abbildung 4.20:** Relativer Fehler und Standardabweichung von Bloomfilter-Array und PCSA in Abhängigkeit von der Anzahl der Sketches bzw. der Größe der verwendeten Arrays.

Abbildung 4.20 stellt den direkten Vergleich der beiden Abschätzungen dar. Aufgrund kleinerer Konfigurationsunterschiede wie beispielsweise der Saat des Zufallsgenerators unterscheiden sich die dargestellten Ergebnisse von PCSA geringfügig von denen in Abbildung 4.3. Der maximale Unterschied beträgt etwas weniger als 0,3 Prozentpunkte und liegt somit in dem zu erwartenden Schwankungsbereich. Für den Vergleich wurde der untersuchte Wertebereich auf zwei Abbildungen mit unterschiedlicher Skalierung aufgeteilt. In dem für PCSA üblichen Wer-

tebereich, welcher auf der linken Seite der Abbildung zu sehen ist, erreicht PCSA durchgehend etwas bessere Ergebnisse als die Abschätzung des BF-Arrays. Ab etwa 2048 Sketches beziehungsweise einer BF-Größe von 2048 erreicht BF-Array bessere Ergebnisse als PCSA und kann den Unterschied im weiteren Verlauf sogar ausbauen. Da aber pro Sketch 16 Bit beziehungsweise immer 16 Arrays der jeweiligen Größe verwendet werden, ist eine derartige Konfiguration nur sinnvoll, wenn eine extrem hohe Genauigkeit erforderlich ist oder sehr viele Elemente gezählt werden müssen. Für die angedachte dezentrale Aggregation ist der Speicherbedarf dieser Konfiguration allerdings eindeutig zu hoch.

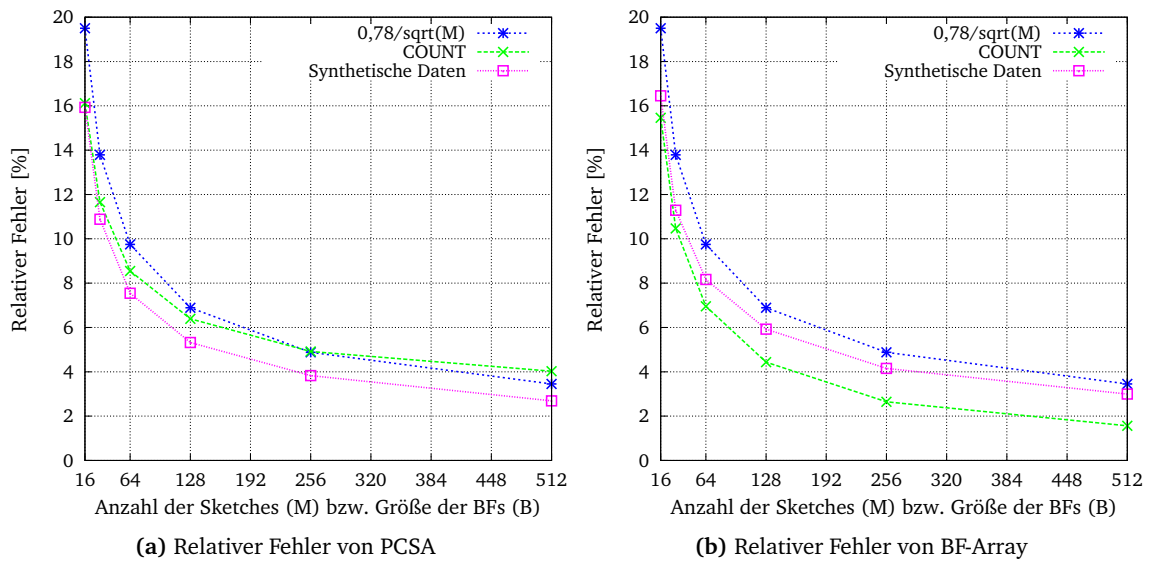


**Abbildung 4.21:** Relativer Fehler von Bloomfilter-Array und PCSA in Abhängigkeit von der Anzahl der eingefügten Elemente.

Die Ergebnisse von BF-Array können möglicherweise durch eine modifizierte Extrapolation der Summe der einzelnen Schätzwerte verbessert werden. Auch kann mit einem etwas höheren Schwellenwert als 80 % die Extrapolation auf eine breitere Datenbasis gestellt werden. Allerdings ist davon auszugehen, dass BF-Array auch mit diesen Verbesserungen keine signifikant höhere Genauigkeit als PCSA erreicht. Der eigentliche Vorteil des Ansatzes liegt in dem Umstand, dass die Schätzung schon nach dem ersten eingefügten Element eine hohe Genauigkeit erzielt. Um diesen Umstand zu verdeutlichen, wurde in Abbildung 4.21 der relative Fehler von BF-Array und PCSA in Abhängigkeit von der Anzahl der eingefügten Elemente dargestellt. Die Ergebnisse von PCSA basieren auf der modifizierten Schätzung, die in Formel 4.4 angegeben ist. Sie erreicht schon mit deutlich weniger Elementen als die ursprüngliche Formel die maximale Genauigkeit. Dennoch ist ein eklatanter Unterschied zwischen den beiden Verfahren zu sehen. Beispielsweise benötigt PCSA bei  $M = 16$  etwa 50 Elemente, um den relativen Fehler auf circa 15 % zu senken, während BF-Array bis zu dieser Anzahl von Elementen sogar bessere Ergebnisse erzielt als im weiteren Verlauf. Da die Genauigkeit nur durch eine Erhöhung der Anzahl der Sketches beziehungsweise durch eine Vergrößerung der Bloomfilter erreicht

werden kann, steigt in diesen Fällen auch die von PCSA benötigte Mindestanzahl an. Wird beispielsweise ein relativer Fehler von um die 3 % benötigt und dementsprechend die Anzahl der Sketches auf 512 erhöht, so steigt auch die Mindestanzahl auf rund 1400 Elemente an.

Im Vergleich zu der Anzahl von Elementen, die maximal gezählt werden können, sind die jeweiligen Mindestanzahlen sehr moderat. Allerdings kann bei einer dezentralen Aggregation weder vorhergesagt werden, wie viele Teilnehmer sich im Beobachtungsgebiet befinden, noch, welche Werte diese in die Datenstruktur einfügen. Doch selbst wenn die mittlere Anzahl in etwa abgeschätzt werden kann, ist es wahrscheinlich, dass aufgrund einer nicht einheitlichen Knotenverteilung starke Schwankungen in der Nachbarschaftsdichte auftreten. Wie stark diese Effekte das Ergebnis beeinflussen, wird im Folgenden mit Hilfe einer Simulation der dezentralen Aggregation untersucht. Der genaue Aufbau und Ablauf der Simulation kann Abschnitt 4.2.3 entnommen werden. Wie zuvor wird die Topologie des Mannheimer Access-Point-Netzwerks aus Abschnitt 5.4 eingesetzt. Als Multiplikator für die COUNT-Funktion wird  $f_c = \text{MIN}(M \cdot 15/12, 400)$  verwendet. Die Funktionen AVG und SUM werden in dieser Simulation nicht berechnet. Da die Mindestanzahl bei einem BF-Array entfällt, wird eine zweite Instanz der Datenstruktur angelegt, bei welcher der Multiplikator nicht berücksichtigt wird.



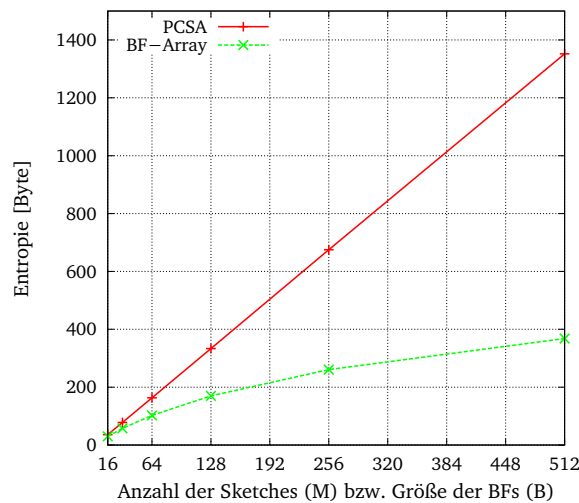
**Abbildung 4.22:** Relativer Fehler der COUNT-Abschätzung von PCSA und BF-Array im Vergleich zu den Ergebnissen mit synthetischen Daten.

Abbildung 4.22 zeigt die Simulationsergebnisse von PCSA und BF-Array im Vergleich zu den jeweiligen Resultaten mit synthetischen Daten. Alle Ergebnisse beziehen sich auf ein Beobachtungsgebiet mit einem Radius von acht Hops. Wie bereits in Abschnitt 4.2.4 liegen auch hier die Ergebnisse der COUNT-Abschätzung von PCSA trotz des großzügig gewählten Multiplikators über den zuvor ermittelten Resultaten mit synthetischen Daten. Im Unterschied dazu verbessern sich die Ergebnisse von BF-Array. Dies liegt an der Tatsache, dass auf Knoten mit unterdurchschnittlich vielen Nachbarn auch auf den Anfangsbereich der beiden Abschätzun-



gen zurückgegriffen werden muss. Da dieser Anfangsbereich bei PCSA eine niedrigere und bei BF-Array eine höhere Genauigkeit aufweist als die jeweilige Abschätzung im weiteren Verlauf, verschlechtert beziehungsweise verbessert sich die durchschnittliche COUNT-Abschätzung des jeweiligen Verfahrens. Dieser Effekt würde sich zwar bei der Berechnung von SUM abschwächen, aber es ist davon auszugehen, dass der relative Fehler auch in diesem Fall nicht über das Niveau der Ergebnisse mit synthetischen Daten hinaus ansteigt.

Die Entropie eines leeren FM-Sketches steigt mit jedem eingefügten Element geringfügig an, bis schließlich die maximale Entropie erreicht wird [134]. Bei  $M = 256$  ist beispielsweise die maximale Entropie nach dem Einfügen von etwa 1024 Elementen erreicht. Auch wenn bei Soft State Sketches aufgrund des größeren Alphabets andere Grenzwerte gelten, kann dort das gleiche Verhalten der Entropie beobachtet werden. Durch den Einsatz eines Multiplikators wird in den bisherigen Simulationen immer der jeweilige Schwellenwert überschritten, sodass trotz gleicher Knotenanzahl und gleichen Sensorwerten die Entropie linear zur Anzahl der Sketches ansteigt (vgl. Abbildung 4.12). Da aber bei einem BF-Array kein Multiplikator benötigt wird, sinkt mit bei einer Erhöhung der Anzahl der Bloomfilter gleichzeitig die Anzahl der Elemente pro Bloomfilter. Falls der jeweilige Schwellenwert unterschritten wird, sinkt damit auch die Entropie der einzelnen Bloomfilter.



**Abbildung 4.23:** Die Entropie von PCSA mit Multiplikator und BF-Array ohne Multiplikator.

Abbildung 4.23 zeigt die durchschnittliche Entropie der simulierten Datenstrukturen von PCSA und BF-Array. Wie zuvor beschrieben, bewirkt der Multiplikator, dass die von PCSA verwendete Datenstruktur immer die maximale Entropie erreicht und mit der Anzahl der Sketches linear ansteigt. Da die Anzahl der Elemente, die in das BF-Array eingefügt werden, immer gleichbleibt, führt hier die Vergrößerung der Bloomfilter nur zu einem stark unterlinearen Wachstum der Entropie.



Die präsentierten Ergebnisse zeigen, dass die Abschätzung von BF-Array im Vergleich zu PCSA die Ergebnisse der dezentralen Aggregation signifikant verbessern kann. Gleichzeitig sinkt durch den Wegfall der Multiplikatoren das Risiko einer suboptimalen oder sogar fehlerhaften Konfiguration. Sofern sich nicht mehrere tausend Knoten im Beobachtungsgebiet befinden, kann beim Bestimmen von COUNT auch noch Speicherplatz eingespart werden. In weniger dichten Netzwerken gilt Gleiches auch für die Abschätzung von SUM. Da BF-Array und PCSA die gleiche Datenstruktur nur unterschiedlich interpretieren, können beide Verfahren auch abschnittsweise miteinander kombiniert werden. Beispielsweise kann die Abschätzung von BF-Array ausschließlich bei nur wenig gefüllten Sketches herangezogen werden. Auf diese Weise können die jeweiligen Vorteile beider Ansätze miteinander verbunden werden. Eine andere denkbare Verbesserung könnte darin bestehen, die Ergebnisse beide Berechnungsansätze zu mitteln oder auf eine andere Weise miteinander zu kombinieren.

## 4.5 Zusammenfassung

In diesem Kapitel wurde die Genauigkeit von PCSA, die bisher nur mathematisch hergeleitet wurde, in einer großen Simulationsstudie im realen Einsatz evaluiert. Dabei konnte gezeigt werden, dass bei synthetischen Eingabedaten sogar eine etwas bessere Genauigkeit erreicht werden konnte, als mathematisch vorhergesagt. Des Weiteren wurde mit PCSA-LT eine neue Abschätzungsmethode vorgestellt, die signifikant bessere Simulationsergebnisse erreichen konnte. Anschließend wurde eine bereits von anderen Autoren vorgeschlagene Strategie zur dezentralen Aggregation im Detail beschrieben und anhand eines großen Netzwerks evaluiert. Im Unterschied zu der Originalpublikation wurde hierbei die Genauigkeit der Datenstruktur in einem größeren Maßstab und ohne externe Abhängigkeiten wie beispielsweise die Netzwerkauslastung oder die Knotenbewegung untersucht. Teil dieser Untersuchung waren auch eine neue Bereichsberechnung des Beobachtungsgebiets und eine eigenständige Kompressionsstrategie. Da sich die Ergebnisse im Vergleich zu den synthetischen Ergebnissen etwas verschlechterten, wurde mit den BF-Maps eine komplett neue Datenstruktur vorgeschlagen, bei der nur die Identifikatoren probabilistisch gespeichert werden. Die präsentierten Simulationsergebnisse zeigen deutlich, dass BF-Maps sogar in dichteren Netzwerken bei einem vergleichbaren Speicherplatzbedarf bessere Ergebnisse erzielt als PCSA und dabei sogar noch die Möglichkeit bietet, jede beliebige Aggregation zu berechnen. Trotz dieser Stärken steigt der Speicherplatz bei sehr großen Beobachtungsgebieten deutlich an, sodass mit BF-Arrays eine weitere alternative Datenstruktur vorgestellt wurde. Zwar unterscheidet sich dieser Ansatz deutlich von PCSA, führt aber zu einer identischen Datenstruktur, sodass die Ergebnisse direkt miteinander verglichen werden können. Bei synthetischen Eingabedaten schneidet PCSA etwas besser ab; allerdings bieten BF-Arrays den Vorteil, dass die Genauigkeit ab dem ersten eingefügten Element erreicht wird. Auf diese Weise konnten bei der dezentralen Aggregation deutlich bessere Ergebnisse als mit PCSA erzielt werden. Da bei BF-Arrays keine Multiplikatoren benötigt werden, vereinfacht sich die Konfiguration, und in weniger dichten Netzwerken

kann sogar Speicherplatz bei der Kompression eingespart werden. Die Ergebnisse zeigen deutlich, dass eine dezentrale Aggregation mit einer probabilistischen Datenstruktur möglich ist und für viele Anwendungsmöglichkeiten eine ausreichende Genauigkeit liefert.

# Kapitel 5

## Integration kooperativer Anwendungen in die bestehende IT-Infrastruktur

Wer das Denken zur Hauptsache macht, der kann es darin zwar weit bringen, aber er hat doch eben den Boden mit dem Wasser vertauscht, und einmal wird er ersaufen.

---

– Hermann Hesse –

Die in Kapitel 3 vorgestellten Routingverfahren sind von der verwendeten Funktechnik unabhängig und können teilweise sogar in drahtgebundenen Netzwerken verwendet werden. Auch der Einsatz der in Kapitel 4 beschriebenen Ansätze zur dezentralen Aggregation ist nicht auf eine bestimmte Netzwerkart oder Übertragungstechnik beschränkt. Aus dieser prinzipiellen Unabhängigkeit folgt aber keineswegs, dass diese Verfahren problemlos in die heute etablierte IT-Infrastruktur integriert werden können. Aus diesem Grund wird in diesem Kapitel erläutert, welche Voraussetzungen gegeben sein müssen und wie diese sinnvoll erfüllt werden können, um die Routingverfahren und Aggregationstechniken integrieren zu können, ohne dass dabei die bestehenden Aufgaben der beteiligten Komponenten signifikant beeinträchtigt werden.

### 5.1 Etablierte Funktechnologien in mobilen Geräten

Grundsätzlich können die präsentierten Verfahren auf allen Computern mit mindestens einer geeigneten Funktechnologie realisiert werden. Da Ad-hoc-Netzwerke allerdings besonders für dynamische Topologien mit mobilen Netzwerkteilnehmern geeignet sind, werden im Folgenden nur tragbare Geräte wie beispielsweise Mobilfunktelefone, Tablet-Computer oder Laptops berücksichtigt. Die in diesen Geräten typischerweise verbauten Funktechniken können in zwei Gruppen eingeteilt werden. Zu der ersten zählen die Mobilfunkstandards wie *Global System for Mobile Communications* (GSM), *Universal Mobile Telecommunications System* (UMTS) oder *Long Term Evolution* (LTE). Die in diesen Standards definierten Funktechniken werden verwendet, um den Clientgeräten den Zugriff auf die von den Mobilfunkbetreibern angebotenen Dienste wie Sprach- und Datenkommunikation zu ermöglichen. Um die bereitgestellte Infrastruktur der Anbieter vor unberechtigt Zugriff zu schützen, sind die in den Clientgeräten verbauten Funkmodule nicht frei konfigurierbar und können auch nicht von privaten Nutzern für eigene

Zwecke modifiziert werden. Des Weiteren sehen die Mobilfunkstandards meist keine direkte Kommunikation zwischen den Clientgeräten vor. Aus diesen Gründen sind die Funktechniken dieser Gruppe nicht für dezentral organisierte und außerhalb der Kontrolle der Mobilfunkanbieter stehende Anwendungen geeignet.

Die zweite Gruppe umfasst alle anderen typischerweise in mobilen Geräten verbauten Funktechniken wie beispielsweise IEEE 802.15.1 *Bluetooth* [79] oder IEEE 802.11 *Wireless Local Area Network* (WLAN) [80]. Ein Hauptzweck von Bluetooth ist es, Kabelverbindungen zwischen Geräten drahtlos zu ersetzen. Beispielsweise kann mit dieser Technologie ein Mobilfunktelefon mit einer Freisprechanlage verbunden werden. In diesem Fall muss zusätzlich zu der reinen Netzwerkebene auch noch definiert werden, welche Daten auf welche Weise zwischen den beiden Geräten ausgetauscht werden. Um diesen Vorgang zu vereinheitlichen, wurden zahlreiche sogenannte Bluetooth-Profile definiert. Beispielsweise kann in dem besprochenen Fall das *Advanced Audio Distribution Profile* (A2DP) [2] für die Sprachübertragung verwendet werden. Verfügt die Freisprechanlage über ein eigenes Display, so kann das *Phone Book Access Profile* (PBAP) [4] eingesetzt werden, um bei eingehenden Anrufen die Telefonnummer oder den im Mobilfunktelefon hinterlegten Namen anzuzeigen. Des Weiteren kann die Freisprechanlage mit Hilfe dieses Profils auch auf das Adressbuch des Telefons zugreifen und so beispielsweise einen Anruf zu einem dort hinterlegten Kontakt initiieren. Neben den zahlreichen Profilen für spezifische Aufgabenstellungen existieren auch generische Profile wie beispielsweise das *Personal Area Networking Profile* (PAN) [3], welches verwendet werden kann, um mehrere Geräte miteinander zu einem sogenannten *Piconet* zu verbinden. Allerdings können in einem derartigen Netzwerk maximal acht Geräte gleichzeitig aktiv sein.

Prinzipiell können neuartige Anwendungen wie die oben beschriebene dezentrale Aggregation von Daten mit Hilfe von Bluetooth realisiert werden. Allerdings müssten hierfür die Limitierungen des PAN-Profiles geschickt umgangen werden. Beispielsweise kann das Netzwerk neben den 8 aktiven bis zu 255 inaktive Geräte enthalten. Zusätzlich können mehrere Piconets zu einem sogenannten *Scatternet* verbunden werden. Eine derartige Vernetzung wird aber durch den Umstand erschwert, dass die meisten Geräte nur wenige Profile unterstützen. Beispielsweise unterstützen Mobilfunktelefone mit Bluetooth häufig nur die thematisch naheliegenden Profile wie etwa den Zugriff auf das Adressbuch und die Sprachübertragung. Aber auch Microsoft Windows XP, Vista und 7 unterstützen von Hause aus nur sechs Profile [10]. Von dem weiter oben erwähnten PAN-Profil unterstützt Microsoft Windows nur die Rolle eines sogenannten PAN-Users. Diese Rolle erlaubt es dem Computer, an einem bestehenden Bluetooth-Netzwerk teilzunehmen oder sich mit einem anderen PAN-User direkt zu verbinden. Daher kann ein Laptop mit Microsoft Windows ohne zusätzliche Hardware oder die Installation eines alternativen - häufig kommerziellen - Bluetooth-Stacks kein Piconet mit mehr als zwei Teilnehmern erzeugen. Statt ein derartiges Netzwerk aufzubauen, könnte auch ein vollständig neues Bluetooth-Profil speziell für dezentrale Anwendungen definiert werden. Alternativ kann der meist frei wählbare Name des Bluetooth-Geräts, der auf Anfrage an die benachbarten Geräte übertragen wird, für den Austausch von kurzen Textnachrichten missbraucht werden,

um die beschriebenen Limitierungen zu umgehen. Allerdings kann mit dieser Vorgehensweise nur eine sehr niedrige Übertragungsgeschwindigkeit erreicht werden. Des Weiteren wird die Nutzung des Bluetooth-Geräts durch die ständigen Namenswechsel möglicherweise beeinträchtigt. Daher kann zusammengefasst werden, dass Bluetooth für die angedachten Zwecke eher ungeeignet ist.

Bei der zweiten Funktechnologie der beschriebenen Gruppe handelt es sich um WLAN. Im Unterschied zu Bluetooth definiert WLAN keine schichtenübergreifenden Anwendungsprofile, sondern beschränkt sich ausschließlich auf die Bitübertragungs- und Sicherungsschicht nach dem OSI-Schichtenmodell beziehungsweise auf die Netzzugangsschicht nach dem TCP/IP-Referenzmodell. WLAN ist daher in erster Linie eine Alternative zu drahtgebundenen Netzwerktechnologien. WLANs können je nach Anwendungszweck in unterschiedlichen Modi betrieben werden. Die zwei wichtigsten werden *Basic Service Set* (BSS) und *Independent Basic Service Set* (IBSS) genannt. Während ein BSS aus einem sogenannten Access Point und beliebig vielen Clients besteht, sind in einem IBSS alle Stationen gleichberechtigt. Auch wenn Access Points häufig als separate Geräte vertrieben werden, ist von der Hardware her jedes WLAN-Gerät in der Lage, als Access Point zu fungieren. Beispielsweise können manche Smartphones als Access Point betrieben werden, um anderen Geräten den Zugriff auf die Internetverbindung über den Mobilfunkbetreiber zu ermöglichen. Im Unterschied zum Infrastrukturmodus werden in einem IBSS die Stationen ad hoc miteinander vernetzt. In diesem Modus kann eine Station mit allen anderen Stationen innerhalb ihrer Funkreichweite direkt kommunizieren. Um auch die Stationen außerhalb der Funkreichweite zu erreichen, muss ein Routingprotokoll einen Pfad zwischen den beiden Kommunikationspartnern finden und die Daten entlang dieses Pfades weiterleiten. Dafür kann beispielsweise das in dieser Arbeit vorgestellte GRANT-Protokoll verwendet werden. Dieser Betriebsmodus von WLAN ist ebenfalls für die in Kapitel 4 beschriebenen kooperativen Anwendungen geeignet.

Sowohl WLAN als auch Bluetooth können so konfiguriert beziehungsweise modifiziert werden, dass sie für die besprochenen Anwendungen verwendet werden können. Da aber der IBSS-Modus im Vergleich zu den benötigten Bluetooth-Profilen eine wesentlich höhere Verbreitung besitzt, ist der zu erwartende Implementierungsaufwand für WLAN wesentlich geringer. Außerdem erreichen die meisten mobilen Bluetooth-Geräte nur eine Funkreichweite von einem bis zu zehn Metern (sogenannte Klasse 1 Geräte). Zwar ist die WLAN-Funkreichweite eines Smartphones geringer als die eines Laptops oder Access Points, aber sie übersteigt die von Bluetooth in der Regel deutlich. Die Verbreitung der beiden Funktechnologien in mobilen Geräten unterscheidet sich nur unwesentlich. Allerdings bietet WLAN in diesem Zusammenhang die interessante Möglichkeit, dass prinzipiell auch die zahlreichen durchgehend aktiven Access Points von den angedachten Anwendungen genutzt werden könnten. Da WLAN aus diesen Gründen die günstigeren Voraussetzungen bietet, wird im Folgenden untersucht, welche Herausforderungen sich bei einem Einsatz in WLANs ergeben und wie diese im Detail gelöst werden können.

### 5.2 Ad-hoc-Netzwerke mit IEEE 802.11 WLAN

Ausgehend von einer geeigneten Konfiguration und einer störungsfreien Übertragung können im IBSS-Modus alle Stationen, die sich innerhalb der Funkreichweite des Senders befinden, das übermittelte Paket empfangen. Daher scheint die Implementierung von neuartigen kooperativen Anwendungen vergleichsweise einfach. In diesem Modus sind auch alle Voraussetzungen dafür erfüllt, ein Ad-hoc-Routingprotokoll für die Ende-zu-Ende-Kommunikation zu verwenden. Doch obwohl das bekannte AODV bereits im Jahre 1999 vorgestellt wurde und zahlreiche Implementierungen dieses Protokolls existieren, wird der Ad-hoc-Modus von WLANs nur sehr selten eingesetzt. Im Folgenden werden die drei wesentlichen Gründe für diese Situation erläutert.

#### Konfiguration

Nach dem Einschalten eines Geräts ist das WLAN typischerweise nicht konfiguriert. Da der klassische Anwendungsfall darin besteht, sich mit einem Access Point zu verbinden, wird meist automatisch eine Suche nach bestehenden Funknetzen in der Umgebung gestartet. Dabei iteriert das Gerät über alle unterstützten Funkkanäle und fragt entweder aktiv nach bestehen Netzwerken oder verweilt etwas länger auf dem jeweiligen Kanal, um die regelmäßig (meist alle 100 ms) verschickten Beacon-Frames der benachbarten Stationen zu empfangen. Nach einem solchen Durchlauf kennt das Gerät die Netzwerke in seiner Umgebung und deren Konfigurationen. Falls kein bekanntes Netzwerk gefunden wird, zu dem automatisch eine Verbindung aufgebaut werden kann, wird dem Benutzer typischerweise eine Auswahlmöglichkeit angezeigt. Um eine Verbindung zu dem gewünschten Netzwerk herzustellen, wechselt das WLAN-Gerät zum jeweiligen Funkkanal und konfiguriert auch die anderen Parameter wie zum Beispiel den *Service Set Identifier* (SSID) automatisch. Falls das gewünschte Netzwerk eine Verschlüsselung nutzen sollte, wird der Benutzer noch nach dem Passwort gefragt. Nach diesen Schritten ist das Gerät dem Netzwerk beigetreten und kann auf physikalischer Ebene mit den anderen Netzteilnehmern in Funkreichweite kommunizieren. Damit allerdings auch auf der Anwendungsebene kommuniziert werden kann, muss in einem IP-basierten Netzwerk dem neuen Gerät noch eine gültige IP-Adresse zugewiesen werden. Aus diesem Grund unterstützen die meisten Access Points das *Dynamic Host Configuration Protocol* (DHCP) [46], welches verwendet werden kann, um dem neuen Gerät automatisch eine eindeutige IP-Adresse zuweisen.

Soll ein Gerät im IBSS-Modus betrieben werden und sich mit einem bestehenden Ad-hoc-Netzwerk verbinden, ist der Ablauf nahezu identisch. Ein wichtiger Unterschied besteht aber in der Zuweisung der IP-Adresse. Im BSS-Modus wird die gesamte Kommunikation von einem Access Point koordiniert. Jede Station kann mit diesem direkt kommunizieren und auf den dort laufenden DHCP-Server zugreifen. Da allerdings in einem Ad-hoc-Netzwerk diese Voraussetzungen nicht sichergestellt werden können, müssen die IP-Adressen dezentral eindeutig vergeben werden. Eine einfache Lösungsmöglichkeit für diese Problematik könnte darin bestehen, dass jede Station einen DHCP-Server betreibt und einen kleinen, aber eindeutigen

Adressbereich zugewiesen bekommt. Alternativ könnten die IP-Adressen auch manuell konfiguriert werden. Da IPv6 einen sehr viel größeren Adressbereich bietet, kann die beschriebene Problematik durch den Einsatz dieser Protokollversion deutlich entschärft werden. Nach einer solchen Konfiguration besitzen zwar alle Stationen eine eindeutige IP-Adresse und befinden sich im gleichen Funknetzwerk, jede Station kann aber trotzdem nur mit den direkt benachbarten Stationen in Funkreichweite kommunizieren. Für eine Multi-Hop-Kommunikation müsste in alle Stationen zusätzlich das gleiche Ad-hoc-Routingprotokoll installiert worden sein. Es existieren zwar zahlreiche Implementierungen von AODV und anderen Protokollen, diese sind aber für Smartphones nicht verfügbar oder nur mit umfangreichem Fachwissen installierbar.

Die zuvor beschriebene Vorgehensweise bezieht sich auf den Verbindungsaufbau zu einem bestehenden Netzwerk. Doch gerade bei einem dezentral organisierten Ad-hoc-Netzwerk muss sich dafür erst einmal eine Station des Netzwerks innerhalb der Funkreichweite befinden. Obwohl es ohne eine derartige Station keinen Kommunikationspartner gibt, kann es dennoch sinnvoll sein, die Konfiguration des WLAN direkt nach dem Einschalten des Geräts vorzunehmen. Beispielsweise können auf diese Weise zwei Stationen, die beide bisher noch nicht mit dem bestehenden Netzwerk verbunden sind, schon miteinander kommunizieren und so eine neue Netzwerkpartition bilden. Außerdem muss letztlich die erste Station manuell konfiguriert werden. Doch genau diese manuelle Konfiguration ist auf den meisten Smartphones entweder gar nicht oder zumindest nicht über die grafische Benutzeroberfläche möglich. Daher kann zusammengefasst werden, dass zwar alle beschriebenen Herausforderungen mit existierenden Technologien lösbar sind, dafür aber häufig die bestehende Software angepasst und erweitert werden müsste. Doch auch nach diesen Anpassungen bleibt die Konfiguration komplex und ist einem Endbenutzer nur eingeschränkt zumutbar. Zusätzlich zu diesen Umständen kommt noch die Problematik hinzu, dass nur wenige WLAN-Treiber den gleichzeitigen Betrieb von IBSS und BBS erlauben. Daher können Geräte ohne einen geeigneten Treiber nicht zum gleichen Zeitpunkt auf ein Ad-hoc-Netzwerk und einen Access Point zugreifen, selbst wenn sich beide auf dem gleichen Funkkanal befinden sollten.

### Skalierbarkeit

Die Skalierbarkeit von Netzwerkprotokollen ist ein wichtiges Bewertungskriterium. Allerdings bezieht sich die per Simulation untersuchte Skalierbarkeit nur auf einen bestimmten Aspekt des Protokolls. Ausgehend von der Anzahl der aktiven Knoten  $n$  kann ein Routingprotokoll beispielsweise dann als skalierbar bezeichnet werden, wenn die Größe der Routingtabellen nicht stärker wächst als  $O(n)$ . Doch wie bereits in der Einleitung zu Kapitel 4 erwähnt, steht jedem Knoten eines Ad-hoc-Netzwerks bei einem zufällig ausgewählten Zielknoten nur eine Bandbreite von  $\Theta(W/\sqrt{n \log n})$  [67] zur Verfügung. Dabei bezeichnet  $n$  weiterhin die Anzahl der Knoten des Ad-hoc-Netzwerks und  $W$  den maximal nutzbaren Datendurchsatz bei einer konstanten Funkreichweite. Diese Limitierung ist ausschließlich durch den grundsätzlichen Aufbau von Ad-hoc-Netzwerken begründet und kann daher auch nicht durch ein an sich skalierbares Routingprotokoll überwunden werden. Um dieser Problematik entgegenzuwirken,

muss das Datenaufkommen mit steigender Entfernung deutlich verringert werden, oder die Pfadlänge muss strikt auf wenige Hops begrenzt werden. Beides ist in einem dezentral organisierten und frei nutzbaren Ad-hoc-Netzwerk kaum zu realisieren. Außerdem können mobile Geräte mit einem Internetanschluss beispielsweise über einen Mobilfunkbetreiber auch ohne Ad-hoc-Vernetzung miteinander kommunizieren. Aus diesen Gründen ist Ad-hoc-Routing eher für geschlossene Netzwerke mit einem spezifischen Anwendungszweck geeignet.

### Nutzen

Die meisten populären Anwendungen basieren auf einer Client-Server-Architektur. Der bei dieser Architektur anfallende Datenverkehr wird ungleichmäßig auf das Netzwerk verteilt und belastet vor allem den Netzbereich um den Server herum. Falls sich sowohl der Server als auch die Clients im Ad-hoc-Netzwerk befinden, wäre die Bandbreite der Knoten in der Nähe des Servers schnell ausgelastet. Daher ist es aus der Sicht des Lastenausgleichs wesentlich sinnvoller, den Server im Internet zu platzieren und die größtenteils ohnehin bereits vorhandenen Internetanschlüsse der mobilen Geräte zu nutzen. Dann allerdings würde das gesamte Ad-hoc-Netzwerk nicht genutzt werden. Ein Kompromiss könnte darin bestehen, dass manche Benutzer ihre Internetverbindung über das Ad-hoc-Netzwerk mit anderen Benutzern in ihrer Umgebung teilen. Aufgrund der im vorherigen Abschnitt erläuterten Skalierungsproblematik und der limitierten Bandbreite von heutigen Mobilfunkverbindungen müsste diese Internetfreigabe aber auf wenige Hops beschränkt werden. Allerdings sind zum jetzigen Zeitpunkt nur Mobilfunktarife erhältlich, die nach Erreichen eines gewissen Datenvolumens die Übertragungsgeschwindigkeit stark drosseln. Des Weiteren kann eine Person, die anderen Zugriff auf ihren Internetanschluss gewährt, auch für die möglicherweise illegalen Handlungen dieser Nutzer haftbar gemacht werden. Daher ist dieses Anwendungsszenario im Augenblick eher unrealistisch.

Neben den Client-Server-Anwendungen gibt es noch andere Möglichkeiten, die direkt via Peer-to-Peer-Verbindung realisiert werden können. Hierzu zählen beispielsweise Chats, der Austausch von Dateien oder Spiele. Doch auch diese Anwendungen sind meist in einen größeren Kontext eingebettet. In einem Stau mag es zum Beispiel für die Beifahrer unterhaltsam sein, sich mit Leidensgenossen in den anderen Fahrzeugen unterhalten zu können, aber in den meisten Fällen soll eine Verbindung zu einer bekannten Person aufgebaut werden. Welche IP-Adresse das Gerät dieser Person zurzeit besitzt und ob dieses Gerät sich überhaupt in der gleichen Netzwerkpartition befindet, muss aber erst ermittelt werden. Daher ist es plausibel, dass selbst bei derartigen Anwendungen häufig ein Teil der Funktionalität auf einen zentralen Server ausgelagert wird. Allerdings kann in einem solchen Szenario auch gleich die Peer-to-Peer-Verbindung über das Internet aufgebaut werden. Aus diesen Gründen bietet ein Ad-hoc-Netzwerk für die bestehenden IP-basierten Anwendungen nur einen geringen zusätzlichen Nutzen.

Die komplexe oder meist gar nicht erst vorhandene Konfigurationsmöglichkeit könnte von den Herstellern der Smartphones leicht durch Softwareaktualisierungen nachgebessert wer-



den. Dabei könnte beispielsweise auch eine herstellerübergreifende Standardkonfiguration bereitgestellt werden. Die Integration eines Ad-hoc-Routingprotokolls ist zwar wesentlich aufwendiger, aber dennoch möglich. Allerdings ist aufgrund der Skalierungsproblematik dieser Netzwerkarchitektur für die Ende-zu-Ende-Kommunikation und des nur geringen zusätzlichen Nutzens für die Anwender davon auszugehen, dass eine derartige Entwicklung auch weiterhin ausbleiben wird.

## 5.3 Beacon-Based Short Message Exchange

Im vorherigen Abschnitt wurden drei wesentliche Ursachen für die geringe Verbreitung von Ad-hoc-Netzwerken im Detail beschrieben. Allerdings gibt es neben den bisher etablierten Anwendungen auch neuartige kooperative Anwendungen, die nicht auf einer klassischen IP-basierten Ende-zu-Ende-Kommunikation aufbauen. In Kapitel 4 wurden mehrere probabilistische Datenstrukturen und Aggregierungsstrategien für derartige Anwendungen beschrieben. Da die Kommunikation bei diesen ausschließlich auf lokalen Broadcasts basiert und kein benachbarter Knoten explizit adressiert werden muss, kann auf eine gültige IP-Konfiguration verzichtet werden. Ebenfalls wird weder ein Routing- noch ein Transportprotokoll benötigt, sodass die Konfiguration deutlich vereinfacht wird. Im Unterschied zu der Ende-zu-Ende-Kommunikation können die ausgetauschten Informationen bei dieser Architektur auf jedem Knoten interpretiert werden. Dadurch kann bei den angedachten kooperativen Anwendungen die Datenmenge mit steigendem Abstand vom Ursprungsort reduziert werden. Des Weiteren ist es möglich, das Verbreitungsgebiet dynamisch an die verfügbare Bandbreite anzupassen. Sollte das Netzwerk stark ausgelastet sein, kann auf diese Weise das Datenaufkommen deutlich reduziert werden, ohne dass der Dienst vollständig eingestellt werden muss. Auch eine Kombination von Ad-hoc-Netzwerk und bestehender Mobilfunkinfrastruktur ist denkbar. Beispielsweise können die Daten von mehreren Geräten lokal aggregiert und dann in anonymisierter Form auch für weiter entfernte Geräte über das Internet bereitgestellt werden. Daher ist es grundsätzlich möglich, die Skalierbarkeit von kooperativen Anwendungen auch in einem frei nutzbaren und offenen Ad-hoc-Netzwerk sicherzustellen.

Die dritte Ursache für die geringe Verbreitung von Ad-hoc-Netzwerken ist der Umstand, dass diese den Benutzern nur wenige Anwendungsmöglichkeiten bieten, die nicht einfacher und zuverlässiger über das Internet realisiert werden können. Wie auch die Skalierungsproblematik zeigt, passt die Ad-hoc-Netzwerkarchitektur schlicht nicht zu den bereits existierenden Anwendungen. Genau an diesem Punkt setzen die angedachten kooperativen Anwendungen an. Der vermeintliche Nachteil der limitierten Funkreichweite wird gewissermaßen als automatischer Interessenbereich genutzt, der in einem klassischen Netzwerk nur durch zusätzlichen Aufwand – beispielsweise die zentrale Analyse von GPS-Daten – nachgebildet werden kann. Ein weiterer Vorteil dieser Architektur ist die Möglichkeit, ohne Paketduplizierung mit allen anderen Geräten in der näheren Umgebung gleichzeitig kommunizieren zu können. Eine einfache, aber nützliche Anwendung könnte beispielsweise darin bestehen, den Benutzer über

die Anwesenheit von Freunden in der näheren Umgebung zu informieren, ohne dafür einem kommerziellen Anbieter den eigenen Aufenthaltsort anvertrauen zu müssen.

Aus den dargestellten Gründen haben kooperative Anwendungen eine Chance, den Endkundenmarkt zu erreichen. Da aber für eine Ad-hoc-Vernetzung per WLAN ohnehin die meisten bestehenden mobilen Geräte wie Smartphones modifiziert werden müssten, sollte diese Veränderung speziell auf den neuen Anwendungszweck abgestimmt werden. Deshalb wird in diesem Abschnitt mit *Beacon-Based Short Message Exchange* (BSMX) [142, 139, 141] eine neue Strategie vorgestellt, mit der kurze Nachrichten per WLAN ausgetauscht werden können. Dabei ist es ein wichtiges Designziel, den primären Verwendungszweck der Geräte so wenig wie möglich zu beeinträchtigen sowie mit dem Nachrichtenaustausch nur eine nützliche und rein optionale Zusatzanwendung zu ermöglichen. Daher wird für diese Kommunikation auch kein einheitlich konfiguriertes Ad-hoc-Netzwerk mit wohldefinierten Eigenschaften benötigt. Stattdessen werden die Nachrichten standardkonform in andere zu versendende WLAN-Frames eingebettet. Werden im aktuellen Betriebsmodus nicht ausreichend viele geeignete Frames verschickt, so besteht auch die Möglichkeit, zusätzliche und ausschließlich zu diesem Zweck bestimmte Frames zu übertragen.

### 5.3.1 Technische Konzeption

Der IEEE-802.11-Standard unterscheidet zwischen sogenannten Daten-, Kontroll- und Managementframes. Jedes Frame kann mit Hilfe eines zusätzlichen Datenfelds, des sogenannten Subtypes, weiter unterteilt werden. Während die Datenframes die Nutzdaten enthalten und an die höheren Netzwerkschichten weitergereicht werden, dienen die Kontrollframes ausschließlich der Koordination des gemeinsamen Medienzugriffs. Dafür werden beispielsweise *Request to Send* (RTS) und *Clear to send* (CTS) Nachrichten ausgetauscht. Managementframes werden zum Verwalten des Netzwerks eingesetzt und ebenfalls nicht an die höheren Netzwerkschichten weitergeleitet. Ein wichtiges Managementframe ist beispielsweise das von Ad-hoc-Stationen oder Access Points typischerweise alle 100 ms übertragene Beacon-Frame. Auch bei der Assoziierung eines Clients mit einem Access Point werden Managementframes ausgetauscht.

Die Datenframes enthalten ein Datenfeld, das den Protokolltyp spezifiziert. Wenn dieser beispielsweise auf 0x86dd gesetzt ist, werden die Nutzdaten des Frames an den IPv6-Stack übergeben. Prinzipiell ist es möglich, für BSMX einen neuen Protokolltyp zu definieren und über diesen Mechanismus Nachrichten auszutauschen. Allerdings reicht der Gerätetreiber normalerweise nur Datenframes weiter, die von demselben Netzwerk stammen (unter anderem gleiche SSIDs) und entweder direkt an diese Station adressiert oder per Broadcast verschickt wurden. Um den angedachten Datenaustausch zu ermöglichen, müsste dieser Mechanismus für die von BSMX verschickten Frames umgangen werden. Des Weiteren soll BSMX auch den Nachrichtenaustausch mit Stationen auf anderen Funkkanälen ermöglichen, wofür ebenfalls der Treiber modifiziert werden muss. Die beschriebene Strategie ist für das Übertragen großer Datenmengen unerlässlich, erfordert aber tiefgreifende Änderungen am Gerätetreiber und am

Betriebssystem. Da bei einem Funkkanalwechsel darauf geachtet werden muss, dass bestehende Verbindungen nicht abbrechen, darf der Wechsel nicht zu lange andauern und muss exakt koordiniert erfolgen. Die benötigten Modifikationen sind deshalb komplex und erfordern hohe Fachkenntnisse, sodass die schnelle Verbreitung eines derartigen Standards nicht zu erwarten ist. Aus diesem Grund wird im Folgenden eine alternative Vorgehensweise beschrieben, die zwar nur für ein geringes Datenvolumen geeignet ist, aber dafür wesentlich einfacher in bestehende Treiber integriert werden kann.

Die grundsätzliche Idee sieht vor, die Nachrichten in bestehende Frametypen einzubetten. Da die Kontrollframes sehr klein sind und ein exaktes Timing benötigen, sind sie für den Transport von zusätzlichen Nachrichten gänzlich ungeeignet. Datenframes wiederum werden nur zum Nutzdatenversand verwendet und besitzen einen vergleichsweise statischen Aufbau. Daher sind die modular aufgebauten Managementframes für eine derartige Modifikation am besten geeignet. Ein weiterer Vorteil ist, dass diese Frames unabhängig vom Nutzdatenaufkommen in vielen Situationen ohnehin übertragen werden und deshalb kein zusätzlicher Paketaustausch initiiert werden muss.

Am besten für den angedachten Anwendungszweck sind Beacon-, Probe-Request- und Probe-Response-Frames geeignet. Diese bestehen größtenteils aus sogenannten *Tagged Parameters*, die in einer sequenziellen Liste gespeichert werden. Jedes Listenelement entspricht einem Tagged Parameter und besteht aus einem Identifikator (ein Byte), einer Längenangabe (ein Byte) und dem eigentlichen Parameter. Die SSID und die unterstützten Datenraten werden beispielsweise als Tagged Parameter mit den Identifikatoren null und eins übertragen. Der Wertebereich der Identifikatoren wird aber nicht voll ausgeschöpft und lässt explizit Platz für zukünftige Erweiterungen. Beispielsweise können Geräteherstellern mit Hilfe spezieller Identifikatoren proprietäre Zusatzfunktionen anbieten. Der implementierte Prototyp von BSMX verwendet den bisher unbenutzten Identifikator 222, um Nachrichten zwischen beliebigen Stationen auszutauschen.

|   |   |                   |
|---|---|-------------------|
| IEEE 802.11 wireless LAN management frame                       |   |                   |
| Fixed parameters (12 bytes)                                     |   |                   |
| Timestamp: 0x000000000994181                                    |   |                   |
| Beacon Interval: 0,102400 [Seconds]                             |   |                   |
| Capability Information: 0x0521                                  |   |                   |
| Tagged parameters (103 bytes)                                   |   |                   |
| SSID parameter set: "BSMX"                                      |   |                   |
| Supported Rates: 1,0(B) 2,0(B) 5,5(B) 11,0(B) 6,0 9,0 12,0 18,0 |   |                   |
| DS Parameter set: Current Channel: 7                            |   |                   |
| Extended Supported Rates: 24,0 36,0 48,0 54,0                   |   |                   |
| Reserved tag number: Tag 222 Len 19                             |   |                   |
| Tag Number: 222 (Reserved tag number)                           |   |                   |
| Tag length: 19  |   |                   |
| Tag interpretation: Not interpreted                             |   |                   |
| 0080  | 00 62 32 2f 00 dd 09 00 03 7f 01 01 00 24 ff 7f | .b2/.... ..\$..   |
| 0090  | de 13 54 68 69 73 20 69 73 20 61 6e 20 65 78 61 | ..This i s an exa |
| 00a0  | 6d 70 6c 65 21 77 21 a8 f8                      | mple!w! . .       |

Abbildung 5.1: Beacon-Frame mit zusätzlicher Textnachricht

Falls ein Gerät als Access Point fungiert oder als gleichberechtigte Station in einem Ad-hoc-Netzwerk eingesetzt wird, können für den Nachrichtenaustausch die im Normalfall alle 100 ms versendeten Beacon-Frames genutzt werden. Abbildung 5.1 zeigt den hinteren Teil eines modifizierten Beacon-Frames. Im oberen Bereich des Bildes sind die sogenannten *Fixed Parameters* wie Zeitstempel und Sendeintervall der Beacon-Frames zu sehen. Dann folgt die Liste mit den üblichen Tagged Parameters wie beispielsweise der SSID. Am Ende der Liste ist die BSMX-Nachricht mit dem Identifikator 222 angehängt. Die Größe einer Nachricht ist durch den Wertebereich der Längenangabe auf 255 Zeichen begrenzt. Prinzipiell ist die Anzahl der Nachrichten pro Beacon-Frame nur durch die maximale Framegröße limitiert. In dem Prototyp wird allerdings immer nur eine Nachricht pro Managementframe verschickt, sodass mit Hilfe von Beacon-Frames in der Standardkonfiguration eine Datenübertragungsrate von circa 2,5 Kilobyte pro Sekunde erreicht werden kann.

Um auch Nachrichten an Stationen auf anderen Funkkanälen übertragen zu können, kann der im Standard vorgesehene aktive Netzwerkscan genutzt werden. Während eines solchen Scans durchläuft das Gerät alle verfügbaren Funkkanäle und sendet jeweils ein sogenanntes Probe-Request-Frame zu allen Access Points der Umgebung, die mit einem Probe-Response-Frame den Empfang bestätigen und Informationen über ihre Konfiguration bereitstellen. Clientgeräte können so während eines Scanvorgangs sowohl Daten zu den Access Points in der Umgebung übertragen, als auch von diesen Daten empfangen. Da während eines Scanvorgangs alle Funkkanäle durchlaufen werden, können die Access Points auf einem beliebigen und dem Client zuvor unbekannten Kanal arbeiten. Die Probe-Request-Frames werden auf jedem Funkkanal per Broadcast an alle Station verschickt, sodass auch Clientgeräte oder Geräte ohne Assoziierung zu einem Access Point diese Frames empfangen können. Daher kann der Scanvorgang als Broadcast über alle Funkkanäle genutzt werden. Die dabei erreichte Datenübertragungsrate ist abhängig von der Verweildauer pro Kanal und vergleichsweise sehr niedrig, kann aber für einige einfache Anwendungen durchaus ausreichend sein.

Da jeder Treiber den beschriebenen Scanvorgang ohnehin implementiert und diesen meist so gestaltet, dass bestehende Verbindungen nicht abbrechen, ist der Modifikationsaufwand vergleichsweise gering. Des Weiteren ist das Risiko für unerwünschte Nebeneffekte im Vergleich zu einer separaten Implementierung eines Kanaldurchlaufs wesentlich geringer. Auch das Anhängen der Nachrichten an die ohnehin übertragenen Beacon-Frames benötigt nur wenige Änderungen am Treiber. Deshalb kann die beschriebene Funktionalität in weniger als hundert Zeilen Quelltext realisiert werden. Falls aber eine hohe Datenübertragungsrate benötigt wird, kann diese nur über die zuvor beschriebene und wesentlich komplexere Spezifikation eines neuen Protokolltyps erreicht werden.

### 5.3.2 Architektur und Implementierung

Der bisher vorgestellte Mechanismus ermöglicht den Datenaustausch durch Managementframes. Damit allerdings die Nutzdaten vom WLAN-Treiber in die Managementframes eingefügt werden können, müssen diese erst einmal in den Treiber gelangen. Für diesen Zweck besitzt der

Prototyp ein Softwaremodul, das in den jeweiligen Linux-basierten WLAN-Treiber eingebunden werden kann. Dieses Modul stellt via *Netlink* (RFC 3549 [132]) eine universelle bidirektionale Schnittstelle bereit. Auf diesem Weg können sowohl zu sendende Daten vom Userspace an den Treiber übermittelt als auch empfangene Daten vom Treiber in den Userspace weitergeleitet werden. Durch das Bündeln aller benötigten Funktionen in der Erweiterung sind im eigentlichen WLAN-Treiber nur noch geringfügige Änderungen notwendig.

Aus Sicherheitsgründen ist es sinnvoll, den Zugriff auf die Schnittstelle zu limitieren, um so zu verhindern, dass Anwendungen ohne Kontrolle Daten an Stationen anderer Netzwerke verschicken. Aus diesem Grund kommuniziert im Prototyp ausschließlich der sogenannte BSMX-Daemon direkt mit dem Kernelmodul. Alle anderen Anwendungen müssen auf die vom Daemon bereitgestellte TCP-Schnittstelle zugreifen. Für die Testumgebung war eine Zugriffskontrolle bisher nicht notwendig. Allerdings kann diese für den produktiven Einsatz leicht in den Daemon integriert werden. Durch diese Architektur ergibt sich des Weiteren der Vorteil, dass die Anwendungen auch auf anderen Rechnern im LAN installiert werden können und nicht wie beim direkten Einsatz von Netlink nur auf das lokale System begrenzt sind. Dieser Umstand ist insbesondere für die meist leistungsschwachen Access Points ohne Monitoranschluss von Bedeutung. Die beschriebene Systemarchitektur wird in Abbildung 5.2 dargestellt.

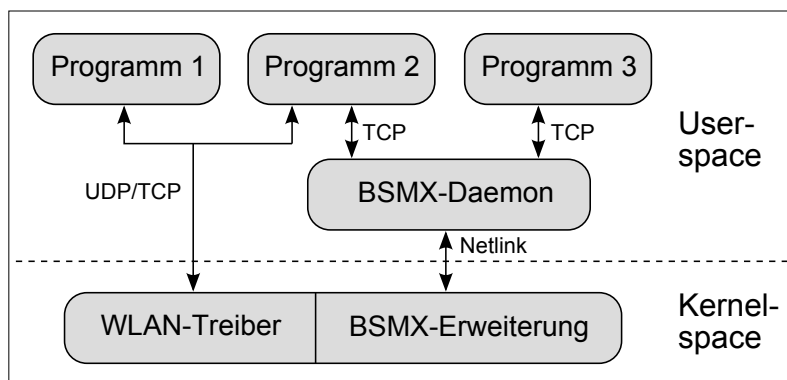


Abbildung 5.2: Die Systemarchitektur von BSMX.

Die von BSMX verschickten Nachrichten sind nicht verschlüsselt und können von allen anderen Stationen mitgelesen werden. Falls ein bestimmter Knoten adressiert werden oder der Netzwerkverkehr verschlüsselt werden soll, muss diese Funktionalität auf der Anwendungsebene implementiert werden. Um die empfangenen Daten einer Anwendung zuordnen zu können, überträgt der BSMX-Daemon vor der Nachricht einen 16-Bit-Zahlenwert, welcher den Nachrichtentyp spezifiziert. Anwendungen können sich beim BSMX-Daemon für den Empfang von bestimmten Nachrichtentypen registrieren. Sind mehrere Anwendungen für den gleichen Nachrichtentyp registriert, so erhalten alle von ihnen beim Eingang einer Nachricht des jeweiligen Typs eine Kopie.

In den BSMX-Daemon können auch die in Kapitel 4 beschriebenen Datenstrukturen und Verbreitungsmechanismen implementiert werden. Auf diese Weise erhalten Anwendungsentwick-

ler den vollen Zugriff auf die verwendeten Datenstrukturen, ohne sich selbst um die komplexe Codierung und Kompression kümmern zu müssen. Im Prototyp wird beispielsweise mit Hilfe von BF-Maps eine Multi-Hop-Nachbarschaftstabelle aufgebaut und aktualisiert. Sobald sich die Nachbarschaftssituation ändert, werden die entsprechenden Anwendungen automatisch informiert. Dafür wird aber nicht die komprimierte BF-Map weitergeleitet, sondern die Nachbarschaft wird in einer leicht decodierbaren Datenstruktur gespeichert. Im nächsten Abschnitt wird eine Beispielanwendung vorgestellt, die diese Informationen visualisiert.

Die BSMX-Erweiterung wurde für Linux implementiert und kann mit modifizierten WLAN-Treibern eingesetzt werden. Exemplarisch wurden die Open-Source-Treiber Madwifi und TNETW modifiziert. Madwifi kann für WLAN-Geräte mit Chipsätzen des Herstellers Atheros verwendet werden und ist beispielsweise für den Einsatz in Linux-basierten Access Points geeignet. TNETW ist Teil des offiziellen Android-SDKs und wird in vielen Smartphones eingesetzt. Der BSMX-Daemon ist in C++ implementiert und kann sowohl für Linux als auch für Android kompiliert werden.

### 5.3.3 Beispielanwendungen

Für den Prototyp standen etwa zwanzig Access Points mit Linux und fünf Smartphones (HTC Hero) mit Android zur Verfügung. Allerdings ist die Implementierung einer großflächigen Aggregation nur sinnvoll, wenn auch entsprechende Sensordaten vorliegen und der gesamte Prototyp im Detail evaluiert wird. Da dies aber die verfügbaren Ressourcen deutlich überstiegen hätte, wurden für Demonstrationszwecke vier einfache Anwendungen realisiert, die auch mit wenigen Geräten sinnvoll einsetzbar sind.

Mit Hilfe der ersten Anwendung können Textnachrichten zwischen Smartphones ausgetauscht werden. Dafür werden die Nachrichten über den BSMX-Daemon in den WLAN-Treiber geladen und dort in ein Probe-Request-Frame eingefügt. Anschließend löst die Anwendung einen Scanvorgang aus, infolgedessen die Nachricht auf allen Funkkanälen verschickt wird. Abbildung 5.3a zeigt ein Screenshot der Anwendung.

Sind zwei Geräte mit dem Internet verbunden, so könnten sie prinzipiell auch über dieses Netzwerk Daten austauschen. Um eine Verbindung aufbauen zu können, muss allerdings mindestens eines der beiden Geräte die IP-Adresse des anderen kennen. Ist der Kommunikationspartner bereits zuvor bekannt, kann die aktuelle IP-Adresse leicht mit Hilfe der Namensauflösung ermittelt werden. Soll aber eine Verbindung zu einem unbekannten Gerät in unmittelbarer Nähe aufgebaut werden, so muss entweder die IP-Adresse selbst oder der Hostname manuell ausgetauscht werden. Die zweite Anwendung vereinfacht diesen Vorgang, indem sie per Knopfdruck die aktuelle IP-Adresse des ausgewählten Netzwerkgeräts zu den benachbarten Knoten überträgt. Auf Wunsch kann die BSMX-Nachricht auch mit dem *Advanced Encryption Standard* (AES) verschlüsselt werden. Der Zielknoten wird hierbei gewissermaßen indirekt durch den Schlüssel adressiert. Abbildung 5.3b zeigt ein Screenshot der Anwendung.

Bei der dritten Anwendung handelt es sich um ein dezentral organisiertes Kartenspiel. In jeder Runde bekommt ein Spieler vier zufällig generierte Karten angezeigt (siehe Abbil-



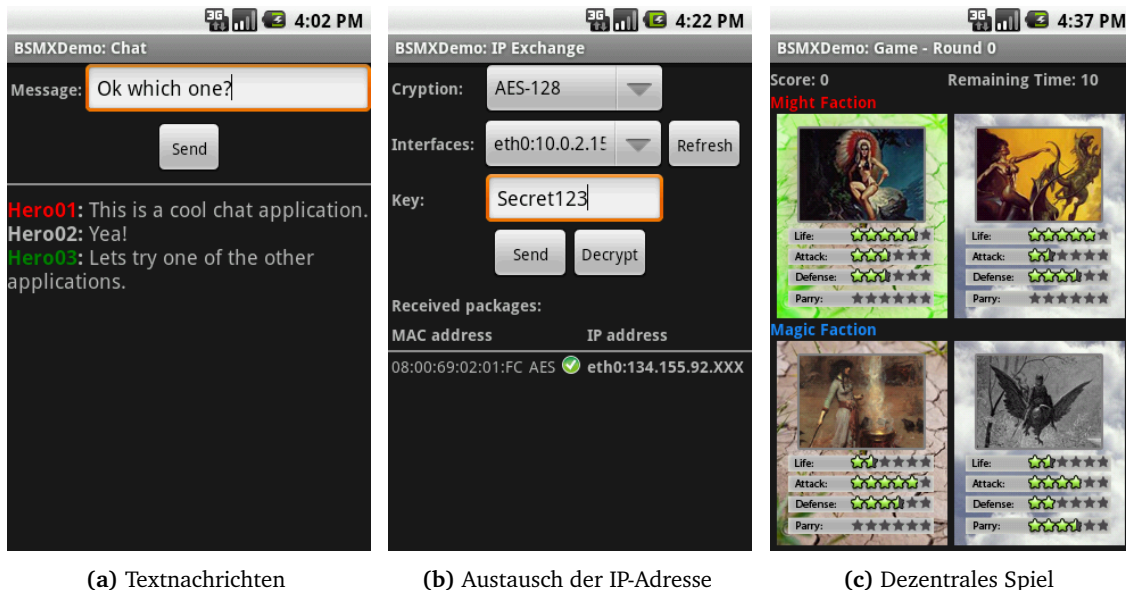


Abbildung 5.3: BSMX: Beispielanwendungen für Android.

dung 5.3c), aus denen er innerhalb der Rundendauer eine Karte auswählen muss. Das Ziel dabei ist es, diejenige Karte auszuwählen, mit der sich der Spieler die besten Chancen gegen seine Mitspieler ausrechnet. Das Prinzip ähnelt dem Spiel *Schere, Stein, Papier*, allerdings besitzt hier jede Karte vier Attribute (Lebenspunkte, Angriff, Verteidigung und Parade). Am Ende der Runde übertragen alle Geräte die jeweils ausgewählten Karten per Broadcast. Sollte ein Spieler innerhalb der Rundenzeit keine Karte ausgewählt haben, setzt er für diese Runde aus. Nach dem Nachrichtenaustausch berechnet jedes Gerät anhand des hinterlegten Algorithmus den Sieger der Runde und zeigt dem Benutzer das Ergebnis an.

Der erste Spieler kann die Rundendauer und den Namen des Spiels festlegen. Anschließend wird diese Konfiguration so lange regelmäßig per Broadcast verschickt, bis ein zweites Gerät dem Spiel beitrifft oder der erste Spieler die Suche abbricht. In den ausgetauschten Nachrichten wird die verbleibende Dauer der nächsten Runde relativ zum Zeitpunkt der Übertragung gespeichert. Anhand dieser Daten und der grob geschätzten Übertragungsdauer kann jeder Empfänger die verbleibende Rundendauer bestimmen. Damit nicht alle Geräte am Ende einer Runde gleichzeitig mit der Übertragung beginnen, wartet jedes Gerät noch eine zufällige Zeit ab. Da die Übertragungen mit Hilfe eines Scanvorgangs erfolgen, liegen die notwendigen Wartezeiten im niedrigen Sekundenbereich. Diese Strategie ist notwendig, um zu vermeiden, dass mehrere Geräte gleichzeitig einen Scan durchführen und so aufgrund der ständigen Kanalwechsel möglicherweise die Frames der anderen Stationen verpassen.

Der BSMX-Daemon verwendet die in Kapitel 4.3 eingeführten komprimierten BF-Maps, um eine Multi-Hop-Nachbarschaftstabelle aufzubauen. Dabei wird die MAC-Adresse des WLAN-Geräts als Identifikator herangezogen, welcher mit dem maximalen TTL-Wert in die Daten-



**Abbildung 5.4:** BSMX: Visualisierung der Nachbarschaftstabelle unter Android und Java Standard Edition.

struktur eingefügt wird. Die vierte Anwendung ermöglicht es dem Benutzer, noch zusätzlich ein Statusbyte in die Datenstruktur einzufügen. Der eigentliche Datenaustausch und die Kombination der empfangenen Datenstrukturen erfolgt gemäß der in Kapitel 4.2 beschriebenen Vorgehensweise. Da in dem Boomfilter einer BF-Map die verbleibende Gültigkeitsdauer des auf die jeweilige Position abgebildeten Identifikators gespeichert wird, kann mit Hilfe der maximalen Gültigkeitsdauer die Entfernung zum jeweiligen Knoten abgeschätzt werden. Die beschriebene vierte Beispielanwendung kann diese Informationen in Form einer Nachbarschaftstabelle visualisieren. Um auch die Access Points mit dem modifizierten Madwifi-Treiber zu testen, wurde diese Anwendung sowohl für Android als auch für die Java Standard Edition entwickelt. Abbildung 5.4 zeigt ein Screenshot von beiden Versionen. Im oberen Bereich der Abbildung sind die Nachbarn in Abhängigkeit von ihrem Abstand in Hops dargestellt. Da die reale Richtung zu den Nachbarn hin unbekannt ist, wird diese für die Visualisierung zufällig bestimmt. Die tabellarische Darstellung der Nachbarschaft inklusive der jeweiligen Statusbytes wird im unteren Bereich der Abbildung dargestellt. Der Prototyp und die beschriebenen Beispielanwendungen wurden auch auf einer gut besuchten Konferenz demonstriert [139].

## 5.4 Empirische Bestimmung der zu erwartenden Konnektivität

Aus der Sicht einer Anwendung ist die Unicast-Kommunikation über eine WLAN-Verbindung mit guten Signalstärken meist zuverlässig, sodass auf der Anwendungsebene üblicherweise keine Paketwiederholungen benötigt werden. Doch selbst unter guten Voraussetzungen treten



häufig Paketkollisionen und -verluste auf, die den höheren Schichten durch Neuübertragungen auf der Sicherungsschicht verborgen bleiben. Per Broadcast verschickte Frames beziehungsweise Pakete wie beispielsweise Beacon-Frames werden auf der Sicherungsschicht nicht bestätigt, sodass ein Paketverlust auch nicht automatisch entdeckt und durch eine Neuübertragung korrigiert werden kann.

Um abschätzen zu können, wie erfolgreich Daten in einem WLAN-basierten Netzwerk mit Hilfe von BSMX ausgetauscht und aggregiert werden können, ist die real zu erwartende Empfangswahrscheinlichkeit von hoher Bedeutung. Allerdings hängt die Wahrscheinlichkeit, ob ein Paket von einem anderen Gerät empfangen werden kann, von vielen Faktoren ab, die unmöglich vollständig erfasst werden können. Insbesondere der Einfluss von externen Faktoren wie der Bebauung der Umgebung und der Kanalauslastung durch unbekannte Geräte kann nicht allgemeingültig modelliert werden. Deshalb soll im Folgenden mit Hilfe einer umfangreichen Messstudie untersucht werden, von welcher Empfangswahrscheinlichkeit im Durchschnitt ausgegangen werden kann.

### 5.4.1 Versuchsaufbau und Durchführung

Befinden sich beide Kommunikationspartner im gleichen Raum beziehungsweise in unmittelbarer Nähe im Außenbereich, so treten Paketverluste typischerweise nur in einem vernachlässigbaren Umfang auf. Sobald sich aber die Entfernung zwischen beiden Geräten vergrößert und möglicherweise noch Hindernisse hinzukommen, steigen die Paketverluste deutlich an. Da die WLAN-Module von mobilen Geräten meist nicht durchgängig aktiv sind und des Weiteren die Gerätepositionen sowie deren Dichte stark von der Tageszeit abhängen, ist es äußerst schwierig, realistische Messungen für derartige Geräte durchzuführen. Daher konzentriert sich die Messung auf die bereits mehrfach erwähnten durchgängig betriebenen Access Points in der Innenstadt von Mannheim. Diese Access Points befinden sich in privaten Wohnungen, sind nicht zugänglich und wurden von den Besitzern beliebig positioniert. Allerdings könnten sie durch eine Aktualisierung ihrer Firmware mit BSMX ausgestattet werden und zukünftig als eine Art Backbone für dezentrale Anwendungen fungieren. Mit der Messung sollte daher untersucht werden, mit welcher Wahrscheinlichkeit die Beacon-Frames der privaten Access Points auf den frei zugänglichen Straßen empfangen werden können.

Der IEEE-802.11b/g-Standard sieht eine Funkkanalbreite von 22 MHz vor. Allerdings betragen die Abstände zwischen den Funkkanälen nur 5 MHz, sodass diese sich teilweise überlappen. Für einen störungsfreien Betrieb wird deshalb empfohlen, nur die Funkkanäle 1, 5, 9 und 13 zu verwenden. Die Konfiguration wird aber den Besitzern der Access Points überlassen, sodass in Wohngebieten mit hoher Wahrscheinlichkeit von Überlappungen auszugehen ist. Aufgrund dieser Überlappungen und der verwendeten Codierung ist es mit einer gewissen Wahrscheinlichkeit möglich, dass ein gesendetes Frame auch von einem Gerät auf einem benachbarten Funkkanal erfolgreich decodiert werden kann. Da solche Frames nicht zum eigenen Funknetzwerk gehören, verwirft der Treiber sie üblicherweise. Allerdings kann BSMX

derartige Beacon-Frames für den Datenaustausch nutzen. Daher soll die Messung auch Aufschluss über die Empfangswahrscheinlichkeiten auf den benachbarten Funkkanälen geben.

Um andere Netzwerke in der Umgebung zu entdecken, sind gemäß dem IEEE-802.11-Standard zwei Verfahren vorgesehen. Beiden Verfahren ist gemeinsam, dass der Client sequenziell durch alle Funkkanäle wechselt und dort jeweils eine bestimmte Zeit auf eingehende Frames wartet. Da Access Points in der Standardeinstellung alle 100 ms per Broadcast ein Beacon-Frame verschicken, kann ein wartender Client durch dessen Empfang über die Existenz des Access Points und seine Konfiguration informiert werden. Dieses Verfahren wird passiver Scan genannt. Im Gegensatz dazu wird bei einem aktiven Scan die Wartezeit minimiert, indem der Client nach jedem Wechsel des Funkkanals ein sogenanntes Probe-Request-Frame verschickt, auf welches die Access Points unmittelbar antworten. Diese beiden Verfahren sind allerdings aufgrund ihres Funktionsprinzips ungeeignet, um die Empfangswahrscheinlichkeiten der Access Points zu bestimmen.

Aus diesem Grund wurden für die Messungen vier Linux-basierte Access Points eingesetzt, die jeweils auf einem vorab konfigurierten Funkkanal alle empfangenen Beacon-Frames aufgezeichnet haben. Auf diese Weise konnten die Funkkanäle zwei, fünf, acht und elf gleichzeitig überwacht werden. Alle Geräte operierten im sogenannten Monitormodus, sodass auch Frames von benachbarten Funkkanälen verarbeitet wurden. Zu Kontrollzwecken wurde ein fünftes Gerät verwendet, das kontinuierlich alle Funkkanäle durchlief und hierbei ebenfalls alle empfangenen Beacon-Frames aufzeichnete. Für die großflächige Messung wurden die fünf Geräte auf dem Dach eines Autos montiert. Während der Messungen wurde auch die jeweils aktuelle Position des Fahrzeugs mit Hilfe GPS aufgezeichnet. Zum Zeitpunkt der Messungen waren neben den reinen Fußgängerzonen auch mehrere Straßenabschnitte aufgrund von Baustellen nicht zugänglich.

### 5.4.2 Abschätzung der Empfangswahrscheinlichkeiten

Insgesamt wurden 3797 unterschiedliche Funknetzwerke in einem Gebiet von circa  $1,42 \text{ km}^2$  entdeckt. Prinzipiell kann ein Access Point mehrere Funknetze mit unterschiedlichen SSIDs gleichzeitig betreiben. In den meisten Fällen kann aber davon ausgegangen werden, dass jedes Funknetzwerk einem Access Point entspricht. Tabelle 5.1 zeigt die am häufigsten verwendeten SSIDs. Der erste Eintrag kombiniert die vorkonfigurierten Namen von vier Gerätevarianten des Herstellers AVM. Diese hohe Verbreitung zeigt, dass möglicherweise auch ein einzelner Anbieter eine ausreichende starke Markposition hat, um einen Ansatz wie BSMX flächendeckend verbreiten zu können.

Die am häufigsten verwendeten Funkkanäle sind eins, sechs und elf, aber auch alle anderen Funkkanäle wurden von jeweils mehr als hundert Access Points benutzt. Das Kontrollgerät mit dem automatischen Funkkanalwechsel konnte nahezu alle von den anderen vier Geräten entdeckten Access Points ebenfalls finden. Die Geräte mit einem vorgegebenen Funkkanal konnten etwa 90-100 % derjenigen Funknetze feststellen, die auf einem benachbarten Kanal betrieben wurden. Des Weiteren konnten circa 60-80 % der Funknetze entdeckt werden,

## 5.4 Empirische Bestimmung der zu erwartenden Konnektivität

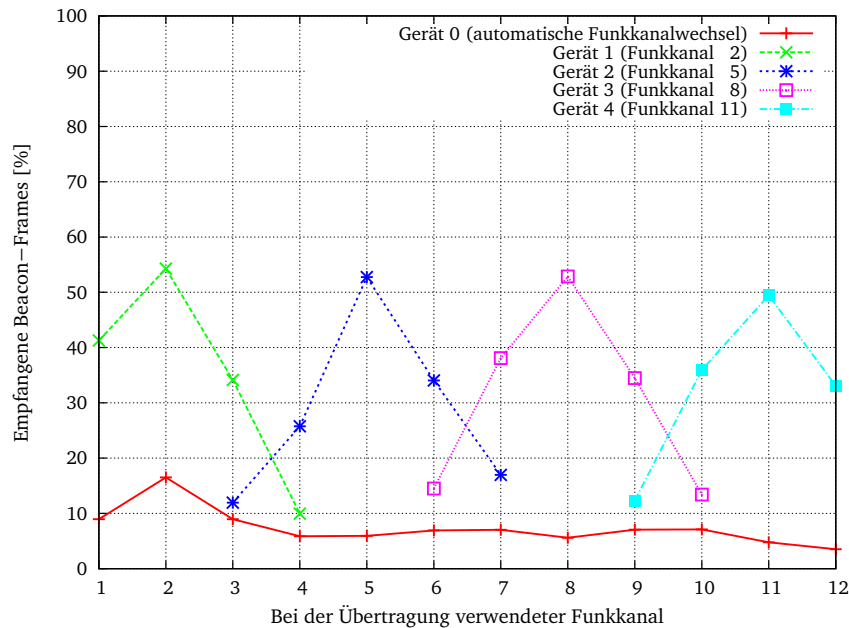
**Tabelle 5.1:** Am häufigsten verwendete SSIDs in der Mannheimer Innenstadt.

| SSID                    | Anzahl | Prozentsatz |
|-------------------------|--------|-------------|
| FRITZ!Box Fon WLAN 7XXX | 434    | 11,43 %     |
| WLAN                    | 90     | 2,37 %      |
| o2DSL                   | 80     | 2,11 %      |
| DSLWLANModem200         | 78     | 2,05 %      |
| default                 | 67     | 1,76 %      |

die zwei Funkkanäle entfernt sendeten. Dafür reicht der Empfang eines einzigen Frames aus. Dementsprechend gibt die Auswertung an, mit welcher Wahrscheinlichkeit ein fahrendes Auto mindestens ein Frame eines passierten Funknetzwerks empfangen kann. Natürlich können bei dieser Berechnung nur Funknetzwerke berücksichtigt werden, die von mindestens einem der fünf Geräte entdeckt wurden.

Um zu bestimmen, mit welcher Wahrscheinlichkeit ein auf Funkkanal  $X$  gesendetes Frame auf Funkkanal  $Y$  im Durchschnitt empfangen werden kann, muss zuvor bekannt sein, wie viele Frames ein Access Point verschickt hat, während sich das Auto in Funkreichweite befand. Da Letztere uneinheitlich und zudem unbekannt ist, muss auf die aufgezeichneten Frames zugegriffen werden. Dafür werden alle von einem Access Point empfangenen Frames als chronologisch sortierte Liste interpretiert. Zwischen dem ersten und dem letzten kontinuierlich empfangenen Frame eines Access Points befand sich das Auto in Funkreichweite. Kontinuierlich bedeutet in diesem Zusammenhang, dass der zeitliche Abstand zwischen zwei aufeinanderfolgenden Frames kleiner als zwei Minuten sein muss; diese Einschränkung ist notwendig, um die Daten von mehreren Besuchen der gleichen Straße voneinander unterscheiden zu können. Durch diese Analyse wird für jeden Access Point der genaue Zeitraum eingegrenzt, in dem sich das Auto in Funkreichweite befunden hat. Die Anzahl der Frames, die während dieses Zeitraums vom jeweiligen Access Point gesendet wurden, wird anhand des verwendeten Beacon-Intervalls bestimmt, das jedem Beacon-Frame entnommen werden kann. Die Ergebnisse aller Access Points des gleichen Funkkanals werden miteinander kombiniert, indem für jedes Messgerät das Verhältnis zwischen allen empfangenen und allen gesendeten Frames dieser Access Points gebildet wird. Abbildung 5.5 zeigt die Ergebnisse für jeden Funkkanal und jedes Messgerät.

Die durchschnittliche Wahrscheinlichkeit, auf der Straße erfolgreich Frames empfangen zu können, die von einem Access Point in einem Gebäude innerhalb der Funkreichweite auf dem gleichen Funkkanal abgesendet wurden, liegt bei circa 53 %. Wie aus der Abbildung zu entnehmen ist, sind die Abweichungen zwischen den vier Geräten mit statischem Funkkanal bei diesem Wert nur gering. Das Gerät auf Funkkanal zwei erreicht mit 54 % die höchste und das Gerät auf Funkkanal elf mit 49 % die niedrigste Empfangsquote. Befindet sich der Absender der Frames auf einem benachbarten Funkkanal, beträgt die durchschnittliche Empfangswahrscheinlichkeit aller Messgeräte mit statischem Funkkanal etwa 35 %. Immerhin noch 14 % aller



**Abbildung 5.5:** Die in der Mannheimer Innenstadt gemessenen Empfangswahrscheinlichkeiten.

Frames konnten empfangen werden, wenn der Access Point auf einem zwei Kanäle entfernten Funkkanal arbeitet. Der verwendete Algorithmus zum automatischen Wechsel des Funkkanals verweilt überproportional lang auf den ersten drei Funkkanälen, sodass das entsprechende Messgerät auf diesen Funkkanälen eine etwas erhöhte Empfangswahrscheinlichkeit aufweist.

Prinzipiell besteht die Möglichkeit, dass die Messgeräte mit statischer Konfiguration von einigen Access Points auf benachbarten Funkkanälen kein Frame empfangen konnten, obwohl dies auf dem jeweiligen Funkkanal des sendenden Access Point möglich gewesen wäre. Ob diese Möglichkeit die bestimmten Wahrscheinlichkeitswerte signifikant beeinflusst, wurde mit Hilfe des Kontrollgerätes überprüft. Dafür wurden zuerst alle vom Kontrollgerät empfangenen Beacon-Frames ausgewählt, die auch auf dem jeweils aktiven Funkkanal des Geräts gesendet wurden. Dann wurden von diesen Frames diejenigen weiter betrachtet, die auf einem Funkkanal abgeschickt wurden, welcher benachbart zu einem der Geräte mit statischer Konfiguration war. Anschließend wurden die jeweiligen Frames von Funkkanal  $X$  mit den empfangenen Frames des Geräts mit dem statischen Funkkanal  $X - 1$  oder  $X + 1$  verglichen. Das Kontrollgerät befand sich beispielsweise für mehrere Sekunden auf dem Funkkanal drei und empfing von einem ebenfalls auf Funkkanal drei arbeitenden Access Point einige Frames. Da das Gerät 1 auf dem benachbarten Funkkanal zwei lauscht, kann getestet werden, wie viele der Frames auch auf Gerät 1 empfangen wurden. Diese Berechnungen ergaben, dass im Durchschnitt circa 63 % dieser Frames auch auf einem Gerät mit statischer Konfiguration empfangen werden konnten. Wie bei allen anderen Geräten kann auch bei dem Kontrollgerät von einer durchschnittlichen Empfangswahrscheinlichkeit von 53 % ausgegangen werden. Aus diesem Umstand und der

vorherigen Analyse folgt, dass auf den benachbarten Funkkanälen circa  $53 \% \cdot 63 \% = 33 \%$  aller Frames empfangen werden konnten. Die zwei Prozentpunkte Unterschied zu dem zuvor bestimmten Wert von  $35 \%$  liegen im Toleranzbereich einer derartigen Messung, sodass davon ausgegangen werden kann, dass der in diesem Absatz besprochene Effekt die zuvor bestimmten Werte nicht signifikant beeinflusst.

### 5.4.3 Konstruktion des Netzwerkgraphen

Die bisherigen Ergebnisse zeigen, dass WLAN-Empfänger auf der Straße ausreichend viele Frames von Access Points empfangen können, um von einem Informationsaustausch zu profitieren. Wenn allerdings die existierenden Access Points als Basisnetz für die dezentrale Aggregation und Verbreitung von Informationen dienen sollen, so ist die zu erwartende Konnektivität zwischen ihnen ebenfalls von großem Interesse. Ohne Zugang zu den Gebäuden ist es allerdings nicht möglich, die Empfangswahrscheinlichkeiten der Access Points untereinander zu messen. Auch aus zeitlichen Gründen kann eine solche Messung nicht für alle Access Points durchgeführt werden. Deshalb wurde ihre Konnektivität anhand der vorhandenen Messungen abgeschätzt und für die Simulation der Informationsverbreitung genutzt.

Um die Empfangswahrscheinlichkeiten zu berechnen, wurden im vorherigen Abschnitt für jeden Access Point und jedes Messgerät das erste und das letzte kontinuierlich empfangene Frame bestimmt. Da in jeder Sekunde auch die Position des Autos gespeichert wurde, ist bekannt, welche Straßenabschnitte zwischen den jeweiligen zwei Frames befahren wurden. Mit diesen Informationen kann für jeden Access Point berechnet werden, auf welchen Straßenabschnitten er mit welcher Wahrscheinlichkeit empfangen werden kann. Die Straßenabschnitte, auf denen ein Access Point empfangen werden kann, können als geometrische Figur interpretiert werden. Mit Hilfe des geometrischen Schwerpunkts dieser Figur kann der Aufstellort des Access Points abgeschätzt werden. Mit Sicherheit wird in einigen Fällen die ermittelte Position deutlich vom realen Aufstellort entfernt sein, aber ohne zusätzliche Informationen ist eine genauere Abschätzung unmöglich. Ein anderes Problem besteht darin, dass sich die ermittelten Orte typischerweise nicht auf einer Straße befinden, sodass für sie auch keine Messdaten vorliegen. Daher werden für die weiteren Auswertungen immer die Messdaten des jeweils am nächsten gelegenen Straßenabschnitts herangezogen.

Abbildung 5.6 zeigt exemplarisch, wie der Netzwerkgraph mit Hilfe der Messdaten konstruiert wurde. Auf den rot markierten Straßenabschnitten kann der Access Point 510 (Funkkanal 6) empfangen werden. Die dem Schwerpunkt am nächsten liegende Messung wurde mit einem roten Pin markiert. Im betrachteten Beispiel befinden sich die angenäherten Positionen von vier anderen Access Points auf den roten Straßensegmenten. Um abzuschätzen, welche dieser vier Access Points Frames von Access Point 510 empfangen können, wird jeweils das Messgerät mit dem kleinsten Abstand im Frequenzbereich herangezogen. So, wie die Messung aufgebaut wurde, kann für jeden Funkkanal  $X$  ein Messgerät ausgewählt werden, das auf  $X - 1$  oder  $X + 1$  konfiguriert war. Im Fall von dem auf Funkkanal sechs arbeitenden Access Point 3673 ist dies das Gerät 2, welches auf Funkkanal fünf Beacon-Frames aufgezeichnet hat. An-

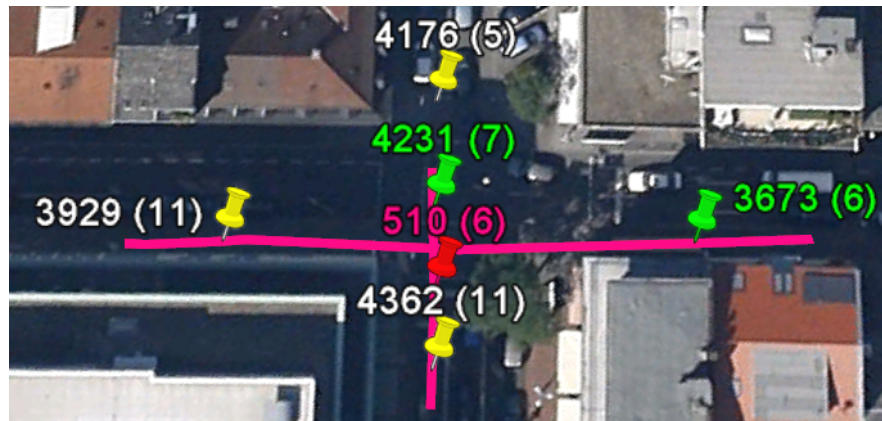


Abbildung 5.6: Konstruktion des Netzwerkgraphen (Screenshot von Google Earth).

hand der Messdaten kann festgestellt werden, dass das Gerät 2 an der Position von Access Point 3673 Frames von Access Point 510 empfangen hat. Dementsprechend wird in dem zu konstruierenden Netzwerkgraphen eine gerichtete Kante von Knoten 510 zu Knoten 3673 erzeugt. An der Position von Access Point 4231 hat das Gerät 3 ebenfalls vereinzelt Frames von Access Point 510 empfangen. An den Positionen der beiden anderen Access Points konnte das jeweilige Messgerät keine Frames von Access Point 510 empfangen.

Für die beiden im Beispiel erzeugten Kanten können auch die jeweiligen Empfangswahrscheinlichkeiten bestimmt werden. Falls das verwendete Messgerät, der empfangende und der sendende Access Point alle auf dem gleichen Funkkanal arbeiten, so kann direkt das Verhältnis zwischen den empfangenen und den gesendeten Frames ermittelt werden. Dies ist aber im Beispiel bei beiden Fällen nicht zutreffend, sodass die mit dem Messgerät bestimmten Wahrscheinlichkeiten noch umgerechnet werden müssen. Hierfür werden die im vorherigen Abschnitt präsentierten Empfangswahrscheinlichkeiten herangezogen. Access Point 3673 wird wie Access Point 510 auf dem Funkkanal sechs betrieben. Wenn auf dem gleichen Funkkanal im Durchschnitt 53 % und auf dem benachbarten Funkkanal 35 % der Frames empfangen werden können, so kann die von Gerät 2 auf dem benachbarten Funkkanal fünf gemessene Wahrscheinlichkeit  $p$  mit  $\text{MIN}(p \cdot (53 \% / 35 \%), 100 \%)$  umgerechnet werden. Im Fall von Access Point 4231 wird die Umrechnung durch den Umstand erschwert, dass dieser Access Point auf Funkkanal sieben arbeitet und das verwendete Messgerät 3 auf Funkkanal acht. Im Unterschied zu dem vorherigen Fall befinden sich somit beide Geräte auf einem anderen Funkkanal als der auf Funkkanal sechs operierende Access Point 510. Das bedeutet, dass sich die gemessene Wahrscheinlichkeit auf eine zwei Funkkanäle entfernte Konfiguration bezieht; sie muss daher erst auf einen benachbarten Funkkanal umgerechnet werden. Im Durchschnitt werden zwei Funkkanäle entfernt 14 % und auf einem benachbarten Funkkanal 35 % der Frames empfangen. Daher muss in diesem Fall die gemessene Wahrscheinlichkeit  $p$  mit  $\text{MIN}(p \cdot (35 \% / 14 \%), 100 \%)$  umgerechnet werden.

Der Netzwerkgraph wird erzeugt, indem die beschriebene Prozedur für alle Access Points wiederholt wird. Dieser Netzwerkgraph wird in Kapitel 4 zur Evaluation der dezentralen Aggregation und im folgenden Abschnitt zur Simulation der Informationsverbreitung verwendet. Die Besonderheit dieses Netzwerks besteht darin, dass aufgrund der uneinheitlichen Konfiguration neben der geografischen Nähe nun auch der Abstand im Frequenzbereich von Bedeutung ist. Beispielsweise benötigen zwei räumlich benachbarte Knoten mit weit auseinanderliegenden Funkkanälen andere Knoten in ihrer Umgebung, die schrittweise den Abstand im Frequenzbereich verringern.

### 5.4.4 Simulation der Informationsverbreitung

Anhand einer Simulationsstudie wird im Folgenden untersucht, ob die zu erwartende Konnektivität zwischen den Access Points ausreicht, um Informationen in einem Innenstadtbereich zu verbreiten. Ebenfalls wird evaluiert, wie viel Zeit für eine derartige Verbreitung benötigt wird.

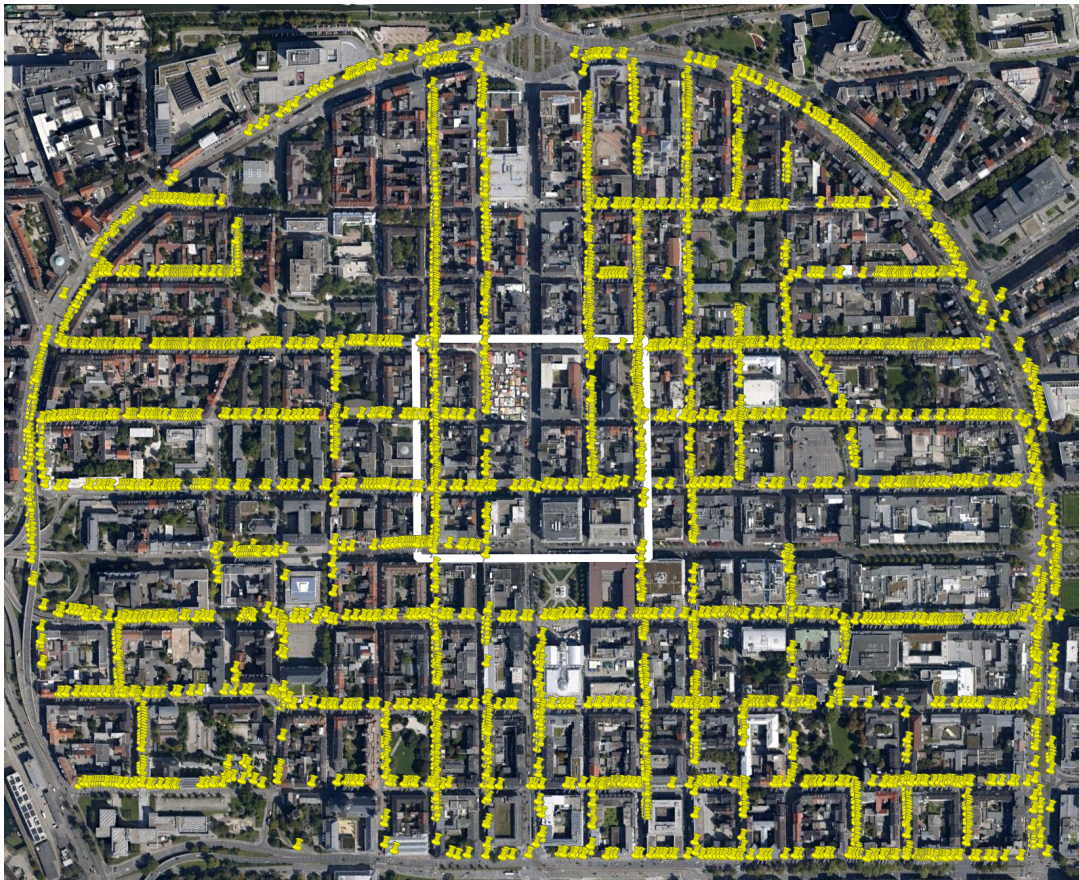
Die zu verbreitende Information ist zu Simulationsbeginn nur einem Knoten bekannt. Dieser verschickt die Information in jeder Simulationsrunde per Broadcast an seine Nachbarn. Der Broadcast wird simuliert, indem das Frame an alle Knoten übermittelt wird, zu denen der Sender eine ausgehende Kante im Netzwerkgraphen besitzt. Um zu entscheiden, ob ein Knoten das abgesendete Frame erfolgreich empfangen hat, wird die an der jeweiligen Position ermittelte Empfangswahrscheinlichkeit herangezogen. Jeder Knoten, der die Information erfolgreich empfangen hat, verschickt diese in den folgenden Runden ebenfalls an seine Nachbarn. Die Simulation wird beendet, sobald jeder erreichbare Knoten die Information kennt oder eine bestimmte Anzahl von Simulationsrunden erreicht wurde. In diesem Zusammenhang ist es wichtig, zu berücksichtigen, dass die Anzahl der erreichbaren Knoten von dem Ausgangsknoten abhängt und daher nicht für alle Simulationsdurchläufe konstant ist.

Ein Knoten am Rand des Simulationsgebiets besitzt eine andere Konnektivität als einer in dessen Zentrum. Deshalb werden in der Simulation als potenzielle Startknoten nur diejenigen Knoten berücksichtigt, die sich innerhalb des in Abbildung 5.7 eingezeichneten Gebiets befinden. Für jeden der 264 Knoten innerhalb der Startzone wurden 100 Simulationsdurchläufe durchgeführt.

Statt des gesamten Netzwerks kann der Simulator auch nur eine Teilmenge der Knoten berücksichtigen und somit eine Informationsverbreitung simulieren, an der sich nur ein bestimmter Prozentsatz der Access Points beteiligt. Auf diese Weise wurde die Informationsverbreitung für Teilnehmerraten zwischen 5 % und 100 % in 5 %-Schritten simuliert. In diesem Fall ist nicht nur die Saat des Zufallsgenerators für die Auswertung der Empfangswahrscheinlichkeiten von Bedeutung, sondern auch die Teilmenge der Knoten, die in der Simulation berücksichtigt wird. Daher wurden für jeden Startknoten die Simulationsergebnisse von 500 Teilnehmerpermutationen kombiniert.

Die Auswertung der Messdaten hat ergeben, dass 97 % aller gefundenen Access Points ein Sendeintervall von 100 ms verwenden. Daher ist es vertretbar, in der Simulation von einem einheitlichen Sendeintervall auszugehen. Dabei verschickt ein informierter Knoten in jeder Si-





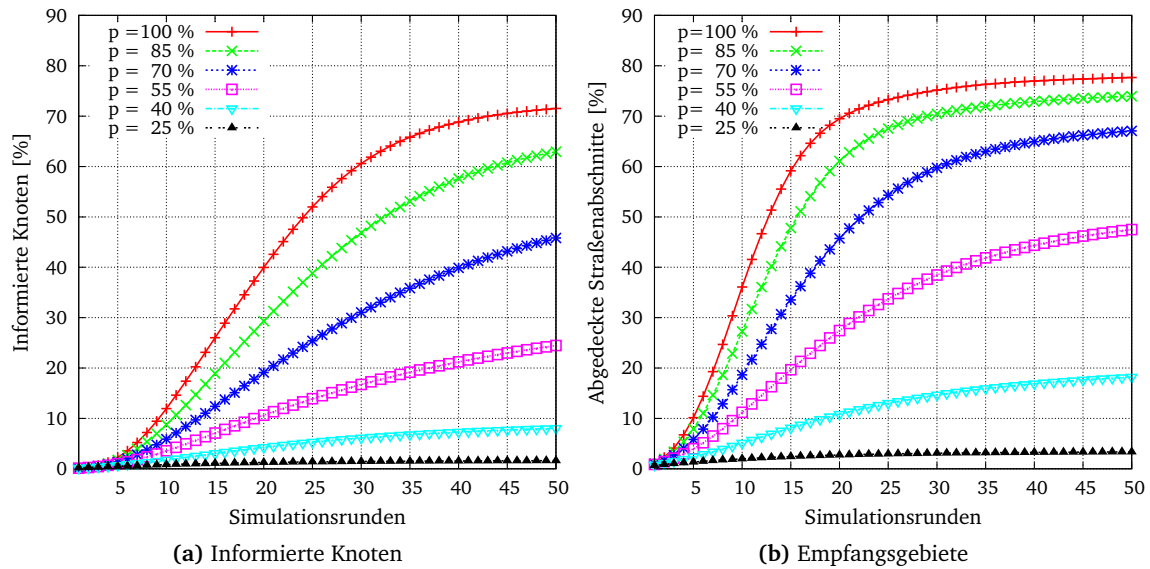
**Abbildung 5.7:** Die entdeckten Access Points in der Mannheimer Innenstadt und die Startzone für die Simulation der Informationsverbreitung.

mulationsrunde ein Beacon-Frame per Broadcast. Im Unterschied zur Simulation schicken die Access Points in der Realität ihre Beacon-Frames allerdings nicht synchronisiert ab. Dementsprechend ist der Zeitraum zwischen dem Empfang eines Beacon-Frames und dem nächsten Sendevorgang im Durchschnitt halb so lang wie das Sendeintervall. Die in der Simulation entstehende Verzögerung ist somit äquivalent zu dem ungünstigsten Fall in einem realen Umfeld. Neben dem vergleichsweise geringen Implementierungsaufwand hat dieses Simulationsmodell den Vorteil, dass die benötigte Rundenanzahl durch die Multiplikation mit der gewünschten Sendeverzögerung in die Echtzeit umgerechnet werden kann, ohne die Simulation erneut durchführen zu müssen.

Abbildungen 5.8a zeigt den Prozentsatz der informierten Knoten und die jeweils benötigte Anzahl von Simulationsrunden für unterschiedliche Teilnehmerraten. Dabei bezieht sich der Prozentsatz der informierten Knoten auf die jeweilige Gesamtanzahl der teilnehmenden Knoten. Falls sich alle Knoten an der Informationsverbreitung beteiligen, kann ein Knoten im Startgebiet im Durchschnitt 75 % der Netzteilnehmer erreichen. Wenn sich jedoch nur 55 %



## 5.4 Empirische Bestimmung der zu erwartenden Konnektivität



**Abbildung 5.8:** Ergebnisse der Simulation der Informationsverbreitung in der Mannheimer Innenstadt.

beteiligen, besteht das Netzwerk aus 2088 Knoten, von denen im Durchschnitt 33 % von der Startzone aus erreichbar sind. Diese Maximalwerte werden allerdings erst bei deutlich mehr als den dargestellten 50 Simulationsrunden erreicht.

Als ein weiteres Kriterium wird die von den informierten Knoten abgedeckte Fläche ausgewertet. In dem betrachteten Szenario sind dies die Straßenabschnitte, auf denen ein Gerät durch einen Netzwerkscan Frames von mindestens einem informierten Access Point empfangen kann. Abbildung 5.8b zeigt den Prozentsatz der besuchten Straßensegmente, auf denen die eingespeiste Information in der jeweiligen Simulationsrunde empfangen werden kann. Wenn sich alle Knoten an der Verbreitung beteiligen, können circa 80 % der besuchten Straßen mit der Information versorgt werden. Bei einer Teilnehmerquote von 55 % kann die Information auf rund 53 % der Straßenabschnitte empfangen werden. Wie bei den vorherigen Ergebnissen werden diese Maximalwerte erst bei mehr als den dargestellten 50 Simulationsrunden erreicht.

Abgesehen von der Anzahl der informierten Knoten und der Größe des Verbreitungsgebiets ist auch die Dauer der Verbreitung von großem Interesse. Wenn sich alle Knoten beteiligen, können in 20 Simulationsrunden circa 40 % von ihnen informiert werden. Unter der Annahme, dass alle Knoten ein Sendeintervall von 100 ms verwenden, entsprechen diese 20 Simulationsrunden 2 Sekunden. Beteiligen sich nur 70 % aller Knoten, so werden 41 Simulationsrunden benötigt, die bei gleichem Sendeintervall 4,1 Sekunden entsprechen. Im Vergleich zur Benachrichtigung der erreichbaren Knoten kann der gleiche Prozentsatz an räumlicher Abdeckung deutlich früher erreicht werden. Wenn beispielsweise 70 % der Knoten die Informationsverbreitung unterstützen, können 40 % der Straßenabschnitte in nur 18 Simulationsrunden er-

reicht werden, während die Benachrichtigung von 40 % der erreichbaren Knoten 41 Simulationsrunden benötigt.

Bei niedrigen Teilnehmerraten kann aufgrund der eingeschränkten Konnektivität nur ein kleiner Teil der anderen Knoten informiert werden. Dabei ist aber zu beachten, dass bei der Simulation die maximal mögliche Reichweite evaluiert wird, die aber womöglich nur von wenigen Anwendungen benötigt wird. Üblicherweise sollte die Verbreitung auf eine Straße oder einen Gebäudeblock limitiert werden. Soll dennoch ein großflächiges Gebiet informiert werden, so können die Access Points die Konnektivität durch temporäre Funkkanalwechsel deutlich erhöhen. Mobile Geräte auf der Straße können ebenfalls helfen, vorhandene Verbindungslücken zwischen den Access Points zu überbrücken. Aus diesen Gründen kann zusammengefasst werden, dass die ermittelte Konnektivität zwischen den Access Points in einer Innenstadtumgebung ausreicht, um mit BSMX Nachrichten auszutauschen.

### 5.5 Zusammenfassung

Zu Beginn dieses Kapitels wurde analysiert, welche der üblicherweise in mobilen Geräten verbauten Funktechnologien für die in dieser Arbeit angedachten Zwecke geeignet sind. Da WLAN die besten Voraussetzungen mit sich bringt, wurde des Weiteren besprochen, welche Herausforderungen es in Bezug auf Konfiguration und Treiber zu beachten gibt. Dabei wurde gezeigt, dass die notwendigen Anpassungen für ein großes Ad-hoc-Netzwerk mit geeignetem Routingprotokoll vergleichsweise umfangreich sind und den normalen Benutzern, die bereits über einen Internetzugang verfügen, nur wenige zusätzliche Vorteile bieten. Daher sind im Endkundenbereich eher kooperative Anwendungen denkbar, die sowohl rein dezentral als auch in Kombination mit dem Internet gestaltet werden könnten. Neben dem höheren Nutzen spricht auch der niedrigere Änderungsaufwand für diese Zielsetzung. Aus diesem Grund wurde mit BSMX eine geschickte Strategie vorgestellt, mit der kurze Nachrichten zwischen nicht assoziierten Geräten ausgetauscht werden können. Dadurch können teilweise auch die Daten von Geräten auf benachbarten Funkkanälen empfangen werden. Des Weiteren kann auf den ohnehin implementierten Kanaldurchlauf des Treibers zurückgegriffen werden, wodurch sich der Änderungsaufwand am Treiber auf ein Minimum reduzieren lässt. Nach der Beschreibung dieses Ansatzes wurden der Prototyp und die zugehörige Softwarearchitektur vorgestellt. Der BSMX-Daemon, der es den anderen Anwendungen ermöglicht, Nachrichten zu verschicken und zu empfangen, implementiert auch die in Kapitel 4.3 vorgestellten BF-Maps und nutzt diese zum Aufbau einer Multi-Hop-Nachbarschaftstabelle. Neben drei einfachen Beispielanwendungen wurde auch eine Anwendung zur Visualisierung dieser Nachbarschaftsinformationen vorgestellt.

Der zweite Teil dieses Kapitels beschreibt eine umfangreiche Messstudie innerhalb der Mannheimer Innenstadt, die durchgeführt wurde, um die Dichte der privaten Access Points in einer derartigen Umgebung herauszufinden und des Weiteren zu bestimmen, mit welcher Wahrscheinlichkeit diese Access Points auf der Straße von Fußgängern oder Fahrzeugen empfangen

werden können. Die Messdaten wurden auch für die Bestimmung der Empfangswahrscheinlichkeiten auf den benachbarten Funkkanälen genutzt. Des Weiteren wurde mit Hilfe der gesammelten Daten ein gerichteter Netzwerkgraph konstruiert, dessen Kantengewichte den jeweils geschätzten Empfangswahrscheinlichkeiten entsprechen. Da die sich daraus ergebende Konnektivität von hoher Bedeutung ist, wurde in diesem Kapitel die reine Informationsverbreitung für unterschiedliche Teilnehmerraten simuliert. Das Hauptergebnis ist die Feststellung, dass ein Netzwerk aus Access Points für eine lokal begrenzte Verbreitung von kompakten Informationen ausreicht und sich die Konnektivität durch temporäre Funkkanalwechsel noch weiter verbessert lässt.



# Kapitel 6

## Zusammenfassung und Ausblick

Jedermann erfindet sich früher oder später  
eine Geschichte, die er für sein Leben hält.

---

– Max Frisch –

Diese Arbeit soll einen kleinen Beitrag dazu leisten, dass eines Tages die inzwischen so zahlreich verbauten Funktechnologien für mehr als allein den direkten Zugriff auf die Infrastruktur verwendet werden. Für diesen Zweck wurden sowohl im Bereich der Routingprotokolle als auch im Bereich der kooperativen Anwendungen neue Verfahren und Ergebnisse präsentiert.

Der Ansatz, bei der Suche nach einem Pfad die Positionen der Knoten zu verwenden, ist in Zeiten von Smartphones und Navigationssystemen in Fahrzeugen mit GPS völlig plausibel. Allerdings sind einige der am häufigsten aufgegriffenen Verfahren wie beispielsweise *Greedy Routing Scheme* (GRS) nur Heuristiken, die keine erfolgreiche Pfadsuche garantieren können, selbst wenn ein Pfad zwischen den beiden Kommunikationspartnern existiert. Dies ist keineswegs eine neue Beobachtung; so wurde beispielsweise bereits die Idee publiziert, GRS mit einer Überbrückungsstrategie wie Face-2 oder mit einer Variante von *Ad hoc On-Demand Distance Vector Routing* (AODV) zu kombinieren. Wie gut diese Zusammenarbeit funktioniert und wie oft überhaupt die Basisstrategie alleine erfolgreich einen Pfad findet, wurde allerdings nicht im Detail untersucht. Da diese Fragestellungen aber von essenzieller Bedeutung sind, um die Eignung der betreffenden Verfahren zu beurteilen, wurde ihnen in Kapitel 3 nachgegangen. Dabei wurde dargelegt, dass die drei wesentlichen Einflussfaktoren für den Erfolg der positionsbasierten Routingprotokolle die Knotendichte, die Knotenverteilung und die Existenz von Funkhindernissen sind. Um diese Analyse zu untermauern, wurde ein spezieller Simulator implementiert, der versucht, von jedem Knoten zu jedem anderen Knoten des Netzwerks einen Pfad zu finden. Sobald in der Simulation die Knotendichte so weit erhöht wurde, dass sich im Durchschnitt mindestens 16 Knoten innerhalb der Funkreichweite befanden, konnte GRS in einem Netzwerk mit gleich verteilten Knoten nahezu jeden Pfad finden. Da das Netzwerk bei deutlich niedrigeren Knotendichten immer stark partitioniert wird, ist es plausibel, für die Evaluation eines Routingprotokolls eine ausreichend hohe Knotendichte zu wählen. Durch eine derartige Konfiguration wird allerdings der Einfluss der Knotendichte auf die Problematik der Pfadsuche nahezu ausgeklammert.

Um die Knotenverteilung als nächsten Einflussfaktor zu untersuchen, wurden die Knoten in den Simulationen nur noch auf den Straßen einer Stadt positioniert. Dafür wurde der Straßenverlauf der Karlsruher Innenstadt digitalisiert. Bereits ab einer Funkreichweite von 300 m und einer ausreichend hohen Knotendichte kann die Ungleichverteilung der Knoten so weit ausgeglichen werden, dass GRS nahezu jeden Pfad findet. Zwar könnten die Knoten noch deutlich ungleichmäßiger auf der Ebene verteilt werden, aber letztlich wäre diese Festlegung völlig willkürlich. Davon abgesehen ist aber ein mehrere Quadratkilometer großes Gebiet ohne jegliche Bebauung höchst unwahrscheinlich, sodass im nächsten Schritt die Umriss der Häuser im Simulationsgebiet anhand von Satellitenaufnahmen digitalisiert und als Funkhindernisse berücksichtigt wurden. Unter diesen Umständen konnte GRS selbst bei einer hohen Knotendichte nur noch für circa 42 % aller möglichen Kommunikationspartner einen Pfad finden. Daher wurden im Folgenden Simulationen mit einer Überbrückungsstrategie durchgeführt, die jedoch aufgrund der schlechten Ergebnisse des Basisverfahrens ebenfalls nicht überzeugen konnten. Aus diesem Grund wurde schließlich die Kreuzungserkennung von *Restricted Greedy Routing* (RGR) näher analysiert, und es wurde untersucht, inwieweit diese genutzt werden kann, um die Leistungsfähigkeit von GRS zu erhöhen. Selbst bei einer fehlerfreien Erkennung aller Kreuzungen konnte nur für etwa 50 % aller Knoten ein Pfad gefunden werden.

Da die beschriebenen Verfahren im Innenstadtgebiet nur unzureichende Ergebnisse lieferten, wurden zu Vergleichszwecken mehrere Verfahren mit zusätzlichem Wissen beschrieben, die alle über 80 % der Pfade gefunden haben. Um das Datenvolumen für die zusätzlichen Informationen zu minimieren, wurde im Anschluss *Greedy Routing with Abstract Neighbour Table* (GRANT) vorgestellt, ein Verfahren, bei dem jeder Knoten in den Beacon-Frames zusätzlich zu seiner eigenen Position auch noch eine als *Abstract Neighbour Table* (ANT) bezeichnete kompakte Version seiner 1-Hop-Nachbarschaftstabelle überträgt. Die Variante von GRANT mit einer 6 Byte großen ANT kann für rund 81 % aller Knoten einen Pfad finden, was eine dramatische Verbesserung gegenüber GRS darstellt. Dennoch wird in einigen Fällen noch eine Überbrückungsstrategie benötigt, sodass sowohl Face-2 als auch eine an AODV angelehnte Strategie ausführlich untersucht wurden. Dabei wurde evaluiert, wie oft eine Überbrückung benötigt wird und wie lang diese im Durchschnitt ist. Des Weiteren wurde im Detail untersucht, wie stark erfolglose Überbrückungsversuche das Netzwerk belasten. Abschließend wurden die Eigenschaften der untersuchten Varianten analysiert und Empfehlungen für unterschiedliche Einsatzszenarien gegeben.

In Kapitel 4 wurde ein vielversprechendes Aggregierungsschema aufgegriffen und im Detail besprochen. Da dieser Ansatz auf *Flajolet-Martin Sketches* (FM-Sketches) aufbaut, wurde als Erstes die Genauigkeit dieser probabilistischen Datenstruktur evaluiert. Dabei konnte gezeigt werden, dass die im Durchschnitt zu erwartenden relativen Fehler sogar etwas unter den mathematisch vorhergesagten Ergebnissen liegen. Anschließend wurde eine im Vergleich zu *Probabilistic Counting with Stochastic Averaging* (PCSA) verbesserte Methode zur Abschätzung präsentiert, die auf einer Lookup-Tabelle aufbaut und deshalb PCSA-LT genannt wird. Im Unterschied zu PCSA werden bei PCSA-LT die Bitmuster der einzelnen Sketches vollständig

---

berücksichtigt. Dadurch kann mit PCSA-LT beispielsweise mit 64 Sketches eine Genauigkeit erreicht werden, die PCSA erst bei 96 Sketches erzielt. Nachdem die grundsätzliche Genauigkeit der FM-Sketches untersucht wurde, lag nun das Augenmerk auf der Fragestellung, welche Genauigkeit die dezentrale Aggregierung mit den sogenannten Soft State Sketches erreichen kann. Dabei zeigte sich, dass PCSA in dieser Simulation etwas schlechtere Ergebnisse erzielte als zuvor im Test mit synthetischen Eingabedaten. Im Unterschied dazu verbesserten sich die durchschnittlichen Ergebnisse von PCSA-LT etwas, wodurch sich der Vorsprung gegenüber PCSA weiter vergrößerte.

Für die Kompression von FM-Sketches wurde bereits ein Vorschlag publiziert, der in dieser Arbeit in modifizierter Form auch für die Soft State Sketches eingesetzt wurde. Daher wurde neben der Genauigkeit auch der im Durchschnitt benötigte Speicherplatz der komprimierten Datenstrukturen untersucht. Die dezentrale Aggregierung erreicht Ergebnisse, welche für zahlreiche Anwendungen ausreichend sind. Allerdings hängt der benötigte Speicherplatz direkt vom Radius des Beobachtungsgebiets und von der Anzahl der Sketches ab. Beispielsweise kann mit PCSA-LT bei 64 Sketches eine Genauigkeit von circa 10 % erreicht werden. Bei einem 8 Hop großen Beobachtungsgebiet benötigt diese Datenstruktur in komprimierter Form im Durchschnitt etwa 171 Byte. Wird die Anzahl der Sketches zur Verbesserung der Genauigkeit auf 256 erhöht, so steigt der Speicherbedarf auf circa 684 Byte. PCSA-LT erreicht bei dieser Konfiguration einen relativen Fehler von etwas mehr als 3 %. Mit diesem Ansatz können nur vergleichsweise einfache Aggregierungen wie COUNT, SUM und AVG bestimmt werden, wobei für die zuletzt genannte bereits zwei Datenstrukturen benötigt werden. Zusätzlich ist die Konfiguration komplex und die Fehleranfälligkeit aufgrund der benötigten Mindestanzahl von eingefügten Elementen hoch. Aus diesen Gründen kann die direkte und damit fehlerfreie Übertragung der Sensorwerte in weniger dichten Netzwerken ein geringeres Datenvolumen benötigen als die probabilistische Datenstruktur mit einer sehr hohen Anzahl von Sketches.

Um in diesen Szenarien eine deutlich höhere Genauigkeit zu erreichen und gleichzeitig die maximale Flexibilität bei der Wahl der Aggregierungsfunktion zu gewährleisten, wurde mit den *Bloomfilter-Maps* (BF-Maps) eine neue Datenstruktur vorgestellt. Deren Besonderheit besteht darin, dass die eigentlichen Sensorwerte verlustfrei gespeichert und nur die Identifikatoren probabilistisch hinterlegt werden. Dafür wird ein Bitarray verwendet, in welchem abhängig vom einzufügenden Identifikator ein bestimmtes Bit gesetzt wird. Wenn beispielsweise ein 8192 Bit großes Array betrachtet wird, beträgt die Entropie bei 1000 gesetzten Bits pro Identifikator nur circa 4,4 Bit. Auf dieser Beobachtung aufbauend werden die realen Identifikatoren wie beispielsweise die 6 Byte großen MAC-Adressen der Knoten auf einem kleineren Wertebereich innerhalb des Bitarrays abgebildet. Dadurch können zwar Abbildungskollisionen auftreten, der benötigte Speicherplatz wird aber gleichzeitig enorm reduziert. Indem anstelle einzelner Bits die entsprechenden TTL-Werte im Array gespeichert werden, kann diese Datenstruktur mit dem gleichen Algorithmus verwendet werden wie die FM-Sketches.

Um die Genauigkeit von BF-Maps zu evaluieren, wurde analog zu den FM-Sketches eine umfangreiche Simulationsstudie durchgeführt. Die Ergebnisse zeigen, dass bei einem Beob-



achtungsgebiet von 4 Hops eine BF-Map mit 384 Zählern inklusive der Sensorwerte im Durchschnitt circa 178 Byte benötigt. Hierbei wird für COUNT beziehungsweise AVG ein relativer Fehler von etwa 2,9 % beziehungsweise 1,6 % erreicht. Im Vergleich dazu benötigt PCSA-LT für eine entsprechende Genauigkeit von COUNT rund 443 Byte, wobei für die Berechnung von AVG dann zwei dieser Datenstrukturen benötigt werden. Um auch die Strategie für sehr dichte und große Beobachtungsgebiete zu verbessern, wurden die *Bloomfilter-Arrays* (BF-Arrays) vorgestellt und gründlich evaluiert. BF-Arrays basieren auf den Überlegungen von BF-Maps und erweitern diese mit einer geometrischen Verteilung, sodass zwar identische Bitmuster wie bei den FM-Sketches entstehen, diese aber spaltenweise und nicht zeilenweise interpretiert werden. Zwar können die BF-Arrays erst ab 2048 Sketches eine höhere Genauigkeit als PCSA erreichen, der Ansatz hat aber den großen Vorteil, dass die Genauigkeit ab dem ersten eingefügten Element zur Verfügung steht und sogar bei wenigen Elementen niedriger ist als im Gesamtdurchschnitt. Dadurch entfallen nicht nur die mit der Mindestanzahl verbundenen Konfigurationsprobleme, sondern es sinkt auch der durchschnittliche relative Fehler bei einer dezentralen Aggregation. So wird beispielsweise mit 256 Sketches ein relativer Fehler von 2,7 % erreicht, wohingegen mit PCSA nur eine Genauigkeit von circa 5 % möglich ist. Da kein Multiplikator benötigt wird, sinkt die Anzahl der Elemente, die im Durchschnitt in einen Sketch eingefügt werden, sodass wiederum die durchschnittliche Entropie sinkt und sich damit auch der benötigte Speicherplatz verringert. Demnach bieten BF-Arrays in einer dezentralen Aggregation eine signifikant höhere Genauigkeit als PCSA und benötigen dabei sogar deutlich weniger Speicherplatz.

Nachdem sowohl die Eignung von positionsbasierten Routingprotokollen als auch die dezentrale Aggregation mit Hilfe probabilistischer Datenstrukturen untersucht wurde, wurde anschließend in Kapitel 5 analysiert, wie derartige Anwendungen am besten in die bestehende IT-Infrastruktur integriert werden können. Dabei zeigte sich, dass WLAN mit dem bestehenden Ad-hoc-Modus die besten Rahmenbedingungen bietet. Allerdings muss für eine erfolgreiche Kommunikation jedes Gerät einheitlich konfiguriert werden. Gerade bei Smartphones sind jedoch nicht alle Konfigurationsoptionen über die normale Benutzeroberfläche zugänglich. Falls der Treiber nicht explizit mehrere sogenannte virtuelle Geräte unterstützt, kann das Gerät im Ad-hoc-Modus nicht gleichzeitig mit einem Access Point verbunden werden. Des Weiteren muss das System auch für die Integration des Routingprotokolls tiefgreifend angepasst werden. Insbesondere bei Smartphones sind normalerweise keine Änderungen am System vorgesehen, sodass die benötigten Anpassungen am besten direkt von den Herstellern der Smartphones beziehungsweise den Autoren des Betriebssystems durchgeführt werden sollten.

Ohne Infrastruktur mit einem entfernten Gerät via Ad-hoc-Routing zu kommunizieren, bietet in Zeiten des Internetzugangs über den Mobilfunk nur in speziellen Anwendungsgebieten einen echten Zusatznutzen. Da die Integration von kooperativen Anwendungen weniger Anpassungen am System erfordert und diese möglicherweise auch bisher ungenutzte Anwendungsmöglichkeiten eröffnen, wurde der Fokus im weiteren Verlauf auf die Integration derartiger Anwendungen gelegt. Mit *Beacon-Based Short Message Exchange* (BSMX) wurde ein

---

Ansatz vorgestellt, der es ermöglicht, Kurznachrichten standardkonform in die Managementframes von WLAN zu integrieren. Auf diese Weise können sowohl die ohnehin regelmäßig verschickten Beacon-Frames als auch der bereits implementierte Kanaldurchlauf zur Suche nach Funknetzwerken genutzt werden, wodurch der Implementierungsaufwand und das Risiko von ungewünschten Nebeneffekten minimiert werden. Des Weiteren bietet dieser Ansatz den Vorteil, dass unabhängig von der aktuellen Konfiguration Nachrichten verschickt werden können und somit das Gerät nahezu ohne Einschränkung verwendet werden kann. Um diese Funktionalität demonstrieren zu können, wurde ein Prototyp implementiert und in dieser Arbeit vorgestellt. Für den Prototyp wurde ein in Access Points weit verbreiteter Treiber modifiziert. Um auch mobile Geräte zu unterstützen, wurde ebenfalls ein Standardtreiber für Smartphones mit Android angepasst. Des Weiteren wurde der BSMX-Daemon implementiert, der als Schnittstelle zu den Userspace-Anwendungen dient und zusätzlich mithilfe von BF-Maps eine Multi-Hop-Nachbarschaftstabelle aufbaut. Zusätzlich wurden auch vier simple Beispielanwendungen implementiert.

Die Besonderheit von BSMX besteht unter anderem darin, dass die kommunizierenden Geräte nicht Teil des gleichen Netzwerks sein müssen. Beispielsweise können zwei voneinander unabhängige Access Points per BSMX Nachrichten austauschen. Befinden sich allerdings beide Access Points nicht auf dem gleichen Funkkanal, so müsste zumindest einer von ihnen für die Kommunikation kurzzeitig den Funkkanal wechseln. Dafür kann zum Beispiel der Kanaldurchlauf der Netzwerksuche herangezogen werden. Da sich die Funkkanäle gemäß des IEEE-802.11b/g-Standards allerdings teilweise überlappen, kann ein Gerät mit einer bestimmten Wahrscheinlichkeit auch Frames von benachbarten Funkkanälen empfangen, sodass in bestimmten Fällen auf einen Kanalwechsel verzichtet werden kann. Um genau zu evaluieren, mit welcher Wahrscheinlichkeit Frames von benachbarten Funkkanälen korrekt decodiert werden können, wurde eine umfangreiche Messstudie in der Mannheimer Innenstadt durchgeführt. Dafür wurde ein Fahrzeug mit fünf Kleinstcomputer ausgestattet, von denen vier auf einem festen Funkkanal konfiguriert waren, während der fünfte kontinuierlich den Funkkanal gewechselt hat. Diese Computer zeichneten alle empfangenen Beacon-Frames auf und speicherten parallel dazu die aktuelle Position des Fahrzeugs, die mittels GPS bestimmt wurde. Um die durchschnittliche Empfangswahrscheinlichkeit zu bestimmen, wurde pro gefundenem Access Point der Zeitpunkt des ersten und des letzten empfangenen Frames bestimmt. Da in jedem Beacon-Frame die Sendefrequenz hinterlegt wird, konnte die Anzahl der Frames, die innerhalb des zuvor bestimmten Zeitfensters abgeschickt wurden, ermittelt werden. Durch einen Vergleich mit der Anzahl der empfangenen Frames konnte dann die Empfangswahrscheinlichkeit für den jeweiligen Access Point geschätzt werden. Da sowohl der Funkkanal des Access Points als auch der Empfangskanal des Überwachungsgeräts bekannt waren, wurden die Einzelergebnisse nach dem Abstand der Funkkanäle gruppiert und dann gemittelt. Im Durchschnitt konnten auf der Straße circa 53 % aller Beacon-Frames, die auf dem gleichen Funkkanal gesendet wurden, erfolgreich empfangen werden. Befand sich der Absender auf einem benach-

barten oder einem zwei Kanäle entfernten Funkkanal, so konnten immerhin noch etwa 35 % beziehungsweise 14 % aller Beacon-Frames empfangen werden.

Abschließend wurde untersucht, ob die zu erwartende Konnektivität zwischen den Access Points für eine Informationsverbreitung ausreicht. Dafür wurde mit Hilfe der Messdaten ein Netzwerkgraph konstruiert, wobei die ermittelten Empfangswahrscheinlichkeiten als Kantengewichte berücksichtigt wurden. Die Simulationsergebnisse zeigen, dass ein Netzwerk aus Access Points für die lokale Verbreitung von kompakten Informationen ausreicht, wenn ein Großteil der Geräte die Weiterverbreitung aktiv unterstützt. Falls nur wenige Access Points teilnehmen, kann die Konnektivität durch temporäre Funkkanalwechsel oder durch die Berücksichtigung der mobilen Geräte verbessert werden. Da die meisten Access Points mit dem Internet verbunden sind, ist auch ein hybrider Ansatz denkbar.

Die in dieser Arbeit präsentierten Ergebnisse könnten größtenteils direkt in bestehende Produkte integriert werden. Beispielsweise könnten ein prominenter Hersteller von Access Points oder ein Anbieter von Internetanschlüssen die Idee von BSMX aufgreifen und in ihre jeweiligen Produkte integrieren. Noch naheliegender wäre eine Integration in die bestehenden Betriebssysteme für Smartphones. Falls zum Beispiel Google oder Apple einen derartigen Ansatz in Android beziehungsweise iOS integrieren würden, würde die resultierende Verbreitung vermutlich für einfache Anwendungen ausreichen.

Zum Abschluss dieser Arbeit wird im Folgenden noch eine Anwendungsidee skizziert. In dem BSMX-Daemon werden BF-Maps für den Aufbau einer Multi-Hop-Nachbarschaftstabelle verwendet. Hierfür berechnet der Daemon den Hashwert der MAC-Adresse des WLAN-Geräts und verwendet anschließend Modulo, um den Hashwert auf einer bestimmten Position des Arrays abzubilden. Da sich der eigene Rechner null Hops weit entfernt befindet, wird an der ermittelten Position des Arrays der maximale TTL-Wert eingetragen. Analog zu der verteilten Aggregation wird dieses Array dann mit Hilfe einer MAX-Operation mit allen empfangenen BF-Maps der Nachbarknoten kombiniert. Anschließend wird in dem resultierenden Array jeder Zähler größer null um eins reduziert. Die komprimierte Datenstruktur wird dann via BSMX zu den benachbarten Knoten übertragen. Die angedachte Weiterentwicklung basiert auf der Beobachtung, dass durch diesen Algorithmus unmittelbar eine Routingtabelle aufgebaut werden kann. Der Absender berechnet dafür anhand der MAC-Adresse des Zielknotens die zugehörige Position in der BF-Map. Anschließend werden die entsprechenden Positionen der empfangenen BF-Maps miteinander verglichen, und das Paket wird an denjenigen Nachbarn übertragen, in dessen BF-Map an dieser Position der höchste TTL-Wert gefunden wurde. Der Empfänger leitet nach der gleichen Regel das Paket weiter, bis schließlich der Zielknoten erreicht wird. Unter der Annahme, dass keine Übertragungsfehler oder sonstige Störungen aufgetreten sind, wird mit dieser Vorgehensweise das Paket automatisch entlang des kürzesten Pfads zum Zielknoten geleitet. Das für die Simulation der dezentralen Aggregation verwendete Netzwerk ist sehr dicht, sodass in den BF-Maps immer vergleichsweise viele Elemente gespeichert werden. Dennoch können diese Ergebnisse als Beispiel herangezogen werden. Jeder Knoten dieses Netzwerks hat im Durchschnitt innerhalb eines Radius von 4 Hops etwa 141 Nachbarn. Wird

---

für dieses Gebiet eine BF-Map mit 4096 Zählern eingesetzt, so sind durchschnittlich 138 Zähler größer null. Das bedeutet, dass im Durchschnitt von etwa drei Knoten der Identifikator durch einen anderen Knoten überschrieben wird. Daher kann in diesem Beispiel mit dem oben skizzierten Routingalgorithmus zu etwa 2 % aller Knoten im Radius von 4 Hops kein korrekter Pfad gefunden werden. Wird der Radius vergrößert, steigt auch die Wahrscheinlichkeit für eine Abbildungskollision, wodurch zu mehr Knoten kein korrekter Pfad gefunden werden kann. Dennoch kann diese Strategie möglicherweise für einige Anwendungen nützlich sein, insbesondere wenn die Routingtabellen als reines Beiwerk einer dezentralen Aggregation kostenlos anfallen. Da die Datenstruktur bei den oben genannten Bedingungen aber im Durchschnitt nur 132 Byte benötigt, kann auch der alleinige Einsatz für das Routing sinnvoll sein. In weniger dichten Netzwerken würde dann auch die Größe der Datenstruktur sinken, wodurch der Radius erhöht werden kann.



# Literaturverzeichnis

## Eigene Publikationen:

[136] [137] [138] [59] [51] [135] [143] [142] [154] [153] [139] [141] [99] [140]

- [1] *AODV Implementation from National Institute of Standards and Technology*. [http://www.antd.nist.gov/wctg/aodv\\_kernel/](http://www.antd.nist.gov/wctg/aodv_kernel/). [Online: Zugriff am 11.12.2010].
- [2] *Bluetooth: Advanced Audio Distribution Profile (A2DP)*. [https://www.bluetooth.org/docman/handlers/DownloadDoc.ashx?doc\\_id=66605](https://www.bluetooth.org/docman/handlers/DownloadDoc.ashx?doc_id=66605). [Online: Zugriff am 20.06.2012].
- [3] *Bluetooth: Personal Area Networking Profile (PAN)*. [https://www.bluetooth.org/docman/handlers/DownloadDoc.ashx?doc\\_id=6554](https://www.bluetooth.org/docman/handlers/DownloadDoc.ashx?doc_id=6554). [Online: Zugriff am 20.06.2012].
- [4] *Bluetooth: Phone Book Access Profile (PBAP)*. [https://www.bluetooth.org/docman/handlers/downloaddoc.ashx?doc\\_id=230887](https://www.bluetooth.org/docman/handlers/downloaddoc.ashx?doc_id=230887). [Online: Zugriff am 20.06.2012].
- [5] *Cornell Research Foundation - JiST / SWANS*. <http://jist.ece.cornell.edu/sw.html>. [Online: Zugriff am 17.02.2011].
- [6] *Creative Commons Attribution-ShareAlike 2.0 Lizenz*. <http://creativecommons.org/licenses/by-sa/2.0/de/legalcode>. [Online: Zugriff am 16.10.2012].
- [7] *Global Mobile Information Systems Simulation Library (GloMoSim)*. <http://pcl.cs.ucla.edu/projects/glomosim/>. [Online: Zugriff am 15.02.2011].
- [8] *International Organization for Standardization - Open Systems Interconnection - Basic Reference Model: The Basic Model (ISO/IEC 7498-1:1994)*. [http://standards.iso.org/ittf/PubliclyAvailableStandards/s020269\\_ISO\\_IEC\\_7498-1\\_1994\(E\).zip](http://standards.iso.org/ittf/PubliclyAvailableStandards/s020269_ISO_IEC_7498-1_1994(E).zip). [Online: Zugriff am 30.06.2011].
- [9] *Keyhole Markup Language (KML)*. <http://code.google.com/apis/kml/>. [Online: Zugriff am 20.11.2011].
- [10] *Microsoft: Bluetooth Wireless Technology FAQ - 2010*. [http://www.microsoft.com/whdc/connect/wireless/Bth\\_FAQ.msp](http://www.microsoft.com/whdc/connect/wireless/Bth_FAQ.msp). [Online: Zugriff am 20.06.2012].
- [11] *The ns-2 Network Simulator*. <http://www.isi.edu/nsnam/ns/>. [Online: Zugriff am 15.02.2011].

- [12] *The ns-3 Network Simulator*. <http://www.nsnam.org/>. [Online: Zugriff am 16.02.2011].
- [13] *OMNeT++ - Network Simulation Framework*. <http://www.omnetpp.org/>. [Online: Zugriff am 15.02.2011].
- [14] *OpenStreetMap*. <http://www.openstreetmap.org/>. [Online: Zugriff am 24.02.2011].
- [15] *Scalable Network Technologie - QualNet*. <http://www.scalable-networks.com/products/qualnet/>. [Online: Zugriff am 16.02.2011].
- [16] *Universität zu Lübeck - Institut für Telematik - Shawn*. <http://shawn.sf.net/>. [Online: Zugriff am 16.02.2011].
- [17] *VISSIM - Multi-Modal Traffic Flow Modeling*. <http://www.vissim.de/>. [Online: Zugriff am 15.02.2011].
- [18] BACKES, W. und J. CORDASCO: *MoteAODV - An AODV Implementation for TinyOS 2.0*. In: *Information Security Theory and Practices: Security and Privacy of Pervasive Systems and Smart Devices*, Bd. 6033 d. Reihe *Lecture Notes in Computer Science*, S. 154–169. Springer Berlin / Heidelberg, 2010.
- [19] BAI, F. und A. HELMY: *Wireless Ad Hoc and Sensor Networks*, Kap. A Survey of Mobility Models, S. 1–30. Springer, October 2006.
- [20] BALAKRISHNAN, H., V. N. PADMANABHAN, S. SESHAN und R. H. KATZ: *A comparison of mechanisms for improving TCP performance over wireless links*. *IEEE/ACM Transactions on Networking (ToN)*, 5(6):756–769, December 1997.
- [21] BELLMAN, R.: *On a Routing Problem*. *Quarterly of Applied Mathematics*, 16:87–90, 1958.
- [22] BETTSTETTER, C., H. HARTENSTEIN und X. PÉREZ-COSTA: *Stochastic properties of the random waypoint mobility model*. *Wireless Networks*, 10(5):555–567, September 2004.
- [23] BLAŽEVIĆ, L., S. GIORDANO und J.-Y. L. BOUDEK: *Self Organized Routing in Wide Area Mobile Ad-hoc Networks*. In: *Proceedings of IEEE Symposium on Ad-Hoc Wireless Networks (Globecom 2001)*, San Antonio, USA, November 2001.
- [24] BLOOM, B. H.: *Space/time trade-offs in hash coding with allowable errors*. *Communications of the ACM*, 13(7):422–426, 1970.
- [25] BLUM, B., T. HE, S. SON und J. STANKOVIC: *IGF: A State-Free Robust Communication Protocol for Wireless Sensor Networks*. Techn. Ber. CS-2003-11, University of Virginia, 2003.
- [26] BONDY, J. A. und U. MURTY: *Graph Theory With Applications*. Elsevier Science Ltd, 1976.

- [27] BOSE, P., P. MORIN, I. STOJMENOVIC und J. URRUTIA: *Routing with guaranteed delivery in ad hoc Wireless Networks*. In: *Proceedings of the 3rd international Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIAL-M 1999)*, S. 48–55, Seattle, USA, August 1999.
- [28] BOUKERCHE, A., H. OLIVEIRA, E. NAKAMURA und A. LOUREIRO: *Localization systems for wireless sensor networks*. *IEEE Wireless Communications*, 14(6):6–12, December 2007.
- [29] BRADEN, R.: *Internet Engineering Task Force - Requirements for Internet Hosts - Communication Layers (RFC 1122)*. <http://tools.ietf.org/html/rfc1122>, October 1989. Status: STANDARD.
- [30] CALISKAN, M., D. GRAUPNER und M. MAUVE: *Decentralized discovery of free parking places*. In: *Proceedings of the 3rd International Workshop on Vehicular Ad Hoc Networks (VANET '06)*, S. 30–39, New York, USA, 2006.
- [31] CAMP, T., J. BOLENG und V. DAVIES: *A Survey of Mobility Models for Ad Hoc Network Research*. *Wireless Communication & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, 2(5):483–502, 2002.
- [32] CAPKUN, S., M. HAMDI und J.-P. HUBAUX: *GPS-free positioning in mobile ad-hoc networks*. *Proceedings of the 34th Annual Hawaii International Conference on System Sciences*, S. 10 pp.–, January 2001.
- [33] CHAKERES, I. D.: *AODV-UCSB Implementation from University of California Santa Barbara*. <http://moment.cs.ucsb.edu/aodv/>. [Online: Zugriff am 11.12.2010].
- [34] CHAKERES, I. D. und E. M. BELDING-ROYER: *AODV Implementation Design and Performance Evaluation*. Special issue on Wireless Ad Hoc Networking of the International Journal of Wireless and Mobile Computing (IJWMC), November/December 2004.
- [35] CHAWLA, M., N. GOEL, K. KALAICHELVAN, A. NAYAK und I. STOJMENOVIC: *Beaconless Position Based Routing with Guaranteed Delivery for Wireless Ad-Hoc and Sensor Networks*. In: *Ad-Hoc Networking*, Bd. 212 d. Reihe *IFIP International Federation for Information Processing*, S. 61–70. Springer Boston, August 2006.
- [36] CHEN, D., J. DENG und P. VARSHNEY: *Selection of a Forwarding Area for Contention-Based Geographic Forwarding in Wireless Multi-Hop Networks*. *IEEE Transactions on Vehicular Technology*, 56(5):3111–3122, September 2007.
- [37] CLARK, B. N., C. J. COLBOURN und D. S. JOHNSON: *Unit disk graphs*. *Discrete Mathematics*, 86(1-3):165–177, 1990.
- [38] CONSIDINE, J., F. LI, G. KOLLIOS und J. BYERS: *Approximate aggregation techniques for sensor databases*. *Proceedings of the 20th International Conference on Data Engineering*, S. 449–460, April 2004.



- [39] CORMEN, T. H., C. E. LEISERSON, R. L. RIVEST und C. STEIN: *Introduction to Algorithms*. MIT Press, 2009.
- [40] CORSON, M. S. und A. EPHREMIDES: *A distributed routing algorithm for mobile wireless networks*. *Wireless Networks*, 1:61–81, February 1995.
- [41] DAS, S., H. PUCHA und Y. HU: *Performance comparison of scalable location services for geographic ad hoc routing*. In: *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2005)*, Bd. 2, S. 1228 – 1239 vol. 2, March 2005.
- [42] DASHTINEZHAD, S., T. NADEEM, B. DOROHONCEANU, C. BORCEA, P. KANG und L. IFTODE: *TrafficView: a driver assistant device for traffic monitoring based on car-to-car communication*. In: *Proceedings of the 59th IEEE Vehicular Technology Conference (VTC '04-Spring)*, Bd. 5, S. 2946 – 2950 Vol.5, May 2004.
- [43] DATTA, S., I. STOJMENOVIC und J. WU: *Internal Node and Shortcut Based Routing with Guaranteed Delivery in Wireless Networks*. *Cluster Computing*, 5:169–178, 2002.
- [44] DELAUNAY, B. N.: *Sur la sphère vide*. *Bulletin of Academy of Sciences of the USSR*, S. 793–800, 1934.
- [45] DIJKSTRA, E. W.: *A note on two problems in connexion with graphs*. *Numerische Mathematik*, 1(1):269–271, 1959.
- [46] DROMS, R.: *Dynamic Host Configuration Protocol (RFC 2131)*. <http://tools.ietf.org/html/rfc2131>, March 1997. Status: DRAFT STANDARD.
- [47] DURAND, M. und P. FLAJOLET: *Loglog Counting of Large Cardinalities*. In: *Proceedings of the 11th Annual European Symposium on Algorithms (ESA 2003)*, 2003.
- [48] ENKELMANN, W.: *FleetNet - applications for inter-vehicle communication*. In: *Proceedings of the IEEE Intelligent Vehicles Symposium 2003*, S. 162 – 167, June 2003.
- [49] FEKETE, S., A. KROLLER, S. FISCHER und D. PFISTERER: *Shawn: The fast, highly customizable sensor network simulator*. In: *Proceedings of the 4th International Conference on Networked Sensing Systems (INSS '07)*, S. 299, June 2007.
- [50] FERGUSON, N., S. LUCKS, B. SCHNEIER, D. WHITING, M. BELLARE, T. KOHNO, J. CALLAS und J. WALKER: *The Skein Hash Function Family*. <http://www.skein-hash.info/sites/default/files/skein1.3.pdf>, 2009. Submission to NIST Cryptographic Hash Algorithm Competition.
- [51] FESTAG, A., G. NOECKER, M. STRASSBERGER, A. LUEBKE, B. BOCHOW, S. SCHNAUFER, M. TORRENT-MORENO, R. EIGNER, C. CATRINESCU und J. KUNISCH: *Network on Wheels: Project Objectives, Technology and Achievements*. In: *Proceedings of the 5th International Workshop on Intelligent Transportation (WIT 2008)*, Hamburg, Germany, February 2008.

- 
- [52] FINN, G. G.: *Routing and addressing problems in large metropolitan-scale internetworks*. Techn. Ber. ISI/RR-87-180, ISI, March 1987.
- [53] FIORE, M., J. HARRI, F. FILALI und C. BONNET: *Vehicular Mobility Simulation for VANETs*. In: *Proceedings of the 40th Annual Simulation Symposium (ANSS '07)*, S. 301–309, March 2007.
- [54] FLAJOLET, P., ÉRIC FUSY, O. GANDOUET und F. MEUNIER.: *HyperLoglog: the analysis of a near-optimal cardinality estimation algorithm*. In: *Proceedings of the Conference on Analysis of Algorithms 2007 (AofA 2007)*, 2007.
- [55] FLAJOLET, P. und G. N. MARTIN: *Probabilistic counting algorithms for data base applications*. *Journal of Computer and System Sciences*, 31(2):182–209, 1985.
- [56] FÜSSLER, H., A. LEIBSCHER, J. WIDMER, M. TRANSIER und W. EFFELSBERG: *Contention-Based Distance-Vector Routing (CBDV) for Mobile Ad-Hoc Networks (Poster)*. In: *Proceedings of the 12th IEEE International Conference on Networking Protocols (ICNP 2004)*, Berlin, Germany, October 2004.
- [57] FÜSSLER, H., M. MAUVE, H. HARTENSTEIN, M. KÄSEMANN und D. VOLLMER: *A Comparison of Routing Strategies for Vehicular Ad Hoc Networks*. Techn. Ber. TR-2002-003, Department for Mathematics and Computer Science, University of Mannheim, July 2002.
- [58] FÜSSLER, H., M. MAUVE, H. HARTENSTEIN, M. KÄSEMANN und D. VOLLMER: *Location-Based Routing for Vehicular Ad-Hoc Networks (Poster)*. In: *Proceedings of the 8th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2002)*, Atlanta, USA, September 2002.
- [59] FÜSSLER, H., S. SCHNAUFER, M. TRANSIER und W. EFFELSBERG: *Vehicular Ad-Hoc Networks: From Vision to Reality and Back*. In: *Proceedings of the 4th Annual Conference on Wireless on Demand Network Systems and Services (WONS 2007)*, S. 80–83, January 2007.
- [60] FÜSSLER, H., J. WIDMER, M. KÄSEMANN, M. MAUVE und H. HARTENSTEIN: *Contention-Based Forwarding for Mobile Ad-Hoc Networks*. Elsevier's *Ad Hoc Networks*, 1(4):351–369, 2003.
- [61] FÜSSLER, H., J. WIDMER, M. MAUVE und H. HARTENSTEIN: *A Novel Forwarding Paradigm for Position-Based Routing (with Implicit Addressing)*. In: *Proceedings of the 18th IEEE Annual Workshop on Computer Communications (CCW 2003)*, S. 194–200, Dana Point, USA, October 2003.
- [62] FORD, L. J. und D. R. FULKERSON: *Flows in Networks*. Princeton University Press, N.J., 1962.
- [63] FORTUNE, S.: *A sweepline algorithm for Voronoi diagrams*. In: *Proceedings of the 2nd Annual Symposium on Computational Geometry, SCG '86*, S. 313–322, New York, USA, 1986.

- [64] FÖRDERER, D.: *Street-Topology Based Routing*. Diplomarbeit, Department of Mathematics and Computer Science, University of Mannheim, May 2005.
- [65] FRIEDMAN, R. und G. KLIOT: *Location Services in Wireless Ad hoc and Hybrid Networks: A Survey*. Techn. Ber. CS-2006-10, Technion - Computer Science Department, April 2006.
- [66] GABRIEL, K. und R. SOKAL: *A new statistical approach to geographic variation analysis*. Systematic Zoology, 18:259–278, 1969.
- [67] GUPTA, P. und P. R. KUMAR: *The Capacity of Wireless Networks*. IEEE Transactions on Information Theory, 46(2):388–404, March 2000.
- [68] HARRI, J., F. FILALI und C. BONNET: *Mobility models for vehicular ad hoc networks: a survey and taxonomy*. IEEE Communications Surveys and Tutorials, 11(4):19–41, 2009.
- [69] HEISSENBÜTTEL, M. und T. BRAUN: *A Novel Position-based and Beacon-less Routing Algorithm for Mobile Ad-Hoc Networks*. In: *Proceedings of the 3rd Workshop on Applications and Services in Wireless Networks (ASWN 2003)*, S. 197–209, Bern, Switzerland, July 2003.
- [70] HEISSENBÜTTEL, M., T. BRAUN, T. BERNOULLI und M. WALCHLI: *BLR: Beacon-Less Routing Algorithm for Mobile Ad-Hoc Networks*. Elsevier Computer Communications, 27(11):1076–1086, 2004.
- [71] HO, C., K. OBRACZKA, G. TSUDIK und K. VISWANATH: *Flooding for reliable multicast in multi-hop ad hoc networks*. In: *Proceedings of the 3rd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIALM '99)*, S. 64–71, New York, USA, 1999.
- [72] HOLLAND, G. und N. VAIDYA: *Analysis of TCP performance over mobile ad hoc networks*. In: *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 1999)*, S. 219–230, New York, USA, August 1999.
- [73] HU, W., V. N. TRAN, N. BULUSU, C. T. CHOU, S. JHA und A. TAYLOR: *The design and evaluation of a hybrid sensor network for Cane-Toad monitoring*. In: *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN '05)*, S. 71, Piscataway, USA, 2005.
- [74] HUANG, H., Y. ZHU, X. LI, M. LI und M.-Y. WU: *META: A Mobility Model of METropolitan TAxis Extracted from GPS Traces*. In: *Proceedings of the IEEE Wireless Communications and Networking Conference 2010 (WCNC 2010)*, S. 1–6, April 2010.
- [75] HUSON, M. und A. SEN: *Broadcast scheduling algorithms for radio networks*. In: *Proceedings of the IEEE Military Communications Conference 1995 (MILCOM '95)*, Bd. 2, S. 647–651 vol.2, November 1995.

- 
- [76] IBRAHIM, K. und M. WEIGLE: *CASCADE: Cluster-Based Accurate Syntactic Compression of Aggregated Data in VANETs*. In: *Proceedings of the IEEE GLOBECOM 2008 (Workshop)*, S. 1–10, December 2008.
- [77] IBRAHIM, K. und M. WEIGLE: *Optimizing CASCADE data aggregation for VANETs*. In: *Proceedings of the 5th IEEE International Conference on Mobile Ad Hoc and Sensor Systems (MASS 2008)*, S. 724–729, October 2008.
- [78] IBRAHIM, K. und M. C. WEIGLE: *POSTER: Accurate data aggregation for VANETs*. In: *Proceedings of the 4th ACM International Workshop on Vehicular Ad Hoc Networks, VANET '07*, S. 71–72, New York, USA, 2007.
- [79] IEEE STANDARD FOR INFORMATION TECHNOLOGY: *Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements - Part 15.1: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Wireless Personal Area Networks (WPANs)*. IEEE Std 802.15.1-2005 (Revision of IEEE Std 802.15.1-2002), 2005.
- [80] IEEE STANDARD FOR INFORMATION TECHNOLOGY: *Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*. IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999), June 2007.
- [81] INTANAGONWIWAT, C., R. GOVINDAN und D. ESTRIN: *Directed diffusion: a scalable and robust communication paradigm for sensor networks*. In: *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MobiCom 2000)*, S. 56–67, New York, USA, 2000.
- [82] JENKINS, B.: *Lookup3*. <http://www.burtleburtle.net/bob/c/lookup3.c>. [Online: Zugriff am 15.06.2012].
- [83] JOHNSON, D. B. und D. A. MALTZ: *Dynamic Source Routing in Ad Hoc Wireless Networks*. In: *Mobile Computing*, Bd. 353 d. Reihe *The International Series in Engineering and Computer Science*, S. 153–181. Springer US, 1996.
- [84] JUANG, P., H. OKI, Y. WANG, M. MARTONOSI, L. S. PEH und D. RUBENSTEIN: *Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with ZebraNet*. *Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'02)*, 37(10):96–107, 2002.
- [85] KAPLAN, E. B.: *Understanding GPS*. Artech House, 1996.
- [86] KARNADI, F., Z. H. MO und K. CHAN LAN: *Rapid Generation of Realistic Mobility Models for VANET*. In: *Proceedings of the IEEE Wireless Communications and Networking Conference 2007 (WCNC 2007)*, S. 2506–2511, March 2007.
- [87] KARP, B. N.: *Challenges in Geographic Routing: Sparse Networks, Obstacles, and Traffic Provisioning*. Talk at DIMACS Workshop on Pervasive Networking, May 2001.

- [88] KARP, B. N. und H. T. KUNG: *GPSR: Greedy Perimeter Stateless Routing for Wireless Networks*. In: *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MobiCom 2000)*, S. 243–254, New York, USA, 2000.
- [89] KIM, M., D. KOTZ und S. KIM: *Extracting a Mobility Model from Real User Traces*. In: *Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM 2006)*, S. 1–13, April 2006.
- [90] KING, T., T. HAENSELMANN, S. KOPF und W. EFFELSBERG: *Positionierung mit Wireless-LAN und Bluetooth*. PIK - Praxis der Informationsverarbeitung und Kommunikation, 29(1):9–17, March 2006.
- [91] KRAJZEWICZ, D., G. HERTKORN, C. RÖSSEL und P. WAGNER: *SUMO (Simulation of Urban MObility) - an open-source traffic simulation*. In: *Proceedings of the 4th Middle East Symposium on Simulation and Modelling (MESM 2002)*, S. 183–187, 2002.
- [92] KRANAKIS, E., H. SINGH und J. URRUTIA: *Compass Routing on Geometric Networks*. In: *Proceedings of the 11th Canadian Conference on Computational Geometry (CCCG 1999)*, S. 51–54, Vancouver, Canada, August 1999.
- [93] KÄSEMANN, M., H. FÜSSLER, H. HARTENSTEIN und M. MAUVE: *A Reactive Location Service for Mobile Ad Hoc Networks*. Techn. Ber. TR-2002-014, Department for Mathematics and Computer Science, University of Mannheim, November 2002.
- [94] KUHN, F., R. WATTENHOFER, Y. ZHANG und A. ZOLLINGER: *Geometric Ad-Hoc Routing: Of Theory and Practice*. In: *Proceedings of the 22nd ACM Annual Symposium on Principles of Distributed Computing (PODC 2003)*, S. 63–72, Boston, USA, July 2003.
- [95] KUHN, F., R. WATTENHOFER und A. ZOLLINGER: *Asymptotically optimal geometric mobile ad-hoc routing*. In: *Proceedings of the 6th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIAL-M 2002)*, S. 24–33, Atlanta, USA, September 2002.
- [96] KULADINITHI, K.: *Java-AODV Implementation from University of Bremen*. <http://www.aodv.org>. [Online: Zugriff am 11.12.2010].
- [97] KUNTZ, A., F. SCHMIDT-EISENLOHR, O. GRAUTE, H. HARTENSTEIN und M. ZITTERBART: *Introducing probabilistic radio propagation models in OMNeT++ mobility framework and cross validation check with NS-2*. In: *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops (Simutools '08)*, S. 72:1–72:7, 2008.
- [98] LEACH, G.: *Improving Worst-Case Optimal Delaunay Triangulation Algorithms*. In: *Proceedings of the 4th Canadian Conference on Computational Geometry*, S. 15, 1992.
- [99] LEMELSON, H., S. SCHNAUFER und W. EFFELSBERG: *Automatic Identification of Fingerprint Regions for quick and reliable Location Estimation*. In: *Proceedings of the 8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM 2010 Workshop Paper)*, S. 540–545, April 2010.

- [100] LEVENBERG, K.: *A method for the solution of certain non-linear problems in least squares*. Quarterly Journal of Applied Mathematics, II(2):164–168, 1944.
- [101] LOCHERT, C.: *Ad Hoc Routing für die Kommunikation zwischen Fahrzeugen in Stadtszenarien*. Diplomarbeit, Department of Mathematics and Computer Science, University of Mannheim, 2003.
- [102] LOCHERT, C., M. CALISKAN, B. SCHEUERMANN, A. BARTHELS, A. CERVANTES und M. MAUVE: *Multiple Simulator Interlinking Environment for Inter Vehicle Communication*. In: *Proceedings of the 2nd ACM International Workshop on Vehicular Ad Hoc Networks (VANET 2005)*, S. 87–88, September 2005.
- [103] LOCHERT, C., H. FÜSSLER, M. MAUVE und H. HARTENSTEIN: *Geographic Routing in City Scenarios*. ACM SIGMOBILE Mobile Computing and Communications Review (MC2R), 9(1):69–72, January 2005.
- [104] LOCHERT, C., H. HARTENSTEIN, J. TIAN, H. FÜSSLER, D. HERRMANN und M. MAUVE: *A Routing Strategy for Vehicular Ad Hoc Networks in City Environments*. In: *Proceedings of IEEE Intelligent Vehicles Symposium 2003 (IV 2003)*, S. 156–161, Columbus, USA, June 2003.
- [105] LOCHERT, C., B. SCHEUERMANN und M. MAUVE: *Probabilistic aggregation for data dissemination in VANETs*. In: *Proceedings of the 4th ACM International Workshop on Vehicular Ad Hoc Networks (VANET 2007)*, S. 1–8, New York, USA, 2007.
- [106] LOCHERT, C., B. SCHEUERMANN und M. MAUVE: *A Probabilistic Method for Cooperative Hierarchical Aggregation of Data in VANETs*. Elsevier Ad Hoc Networks, 8(5):518–530, July 2010.
- [107] MADDEN, S., M. J. FRANKLIN, J. M. HELLERSTEIN und W. HONG: *TAG: a Tiny AGgregation service for ad-hoc sensor networks*. ACM SIGOPS Operating Systems Review, 36(SI):131–146, 2002.
- [108] MAINWARING, A., D. CULLER, J. POLASTRE, R. SZEWCZYK und J. ANDERSON: *Wireless sensor networks for habitat monitoring*. In: *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications (WSNA 2002)*, S. 88–97, New York, USA, 2002.
- [109] MATSUMOTO, M. und T. NISHIMURA: *Mersenne twister: a 623-dimensionally equi-distributed uniform pseudo-random number generator*. ACM Transactions on Modeling and Computer Simulation, 8:3–30, January 1998.
- [110] NADEEM, T., S. DASHTINEZHAD, C. LIAO und L. IFTODE: *TrafficView: a scalable traffic monitoring system*. In: *Proceedings of the 5th IEEE International Conference on Mobile Data Management (MDM 2004)*, S. 13 – 26, January 2004.
- [111] NAKAGAMI, N.: *The m-distribution, a general formula for intensity distribution of rapid fading*. In: HOFFMAN, W. G. (Hrsg.): *Statistical Methods in Radio Wave Propagation*. Oxford, England: Pergamon, 1960.

- [112] NARASAWA, M., M. ONO und H. HIGAKI: *NB-FACE: No-Beacon FACE Ad-hoc Routing Protocol for Reduction of Location Acquisition Overhead*. In: *Proceedings of the 7th International Conference on Mobile Data Management (MDM 2006)*, S. 102 – 102, May 2006.
- [113] NASIPURI, A. und S. DAS: *On-demand multipath routing for mobile ad hoc networks*. *Proceedings of the 8th International Conference on Computer Communications and Networks*, S. 64–70, 1999.
- [114] NATIONAL IMAGERY AND MAPPING AGENCY: *Department of Defense World Geodetic System 1984: Its Definition and Relationships With Local Geodetic Systems*. Techn. Ber. TR8350.2, National Imagery and Mapping Agency, 2000.
- [115] NAVIDI, W. und T. CAMP: *Stationary distributions for the random waypoint mobility model*. *IEEE Transactions on Mobile Computing*, 3(1):99 – 108, February 2004.
- [116] NI, S.-Y., Y.-C. TSENG, Y.-S. CHEN und J.-P. SHEU: *The broadcast storm problem in a mobile ad hoc network*. In: *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '99)*, MobiCom '99, S. 151–162, New York, USA, 1999.
- [117] NORDSTROM, E.: *AODV-UU Implementation from University of Uppsala*. <http://aodvu.sourceforge.net/>. [Online: Zugriff am 11.12.2010].
- [118] OKABE, A., B. BOOTS, K. SUGIHARA und S. N. CHI: *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. Wiley, Chichester, UK, second Aufl., 2000.
- [119] PATEL, M.: *The Goulburn Hashing Function*. <http://www.webcitation.org/query?url=http://www.geocities.com/drone115b/Goulburn06.pdf>. [Online: Zugriff am 15.06.2012].
- [120] PERKINS, C., E. BELDING-ROYER und S. DAS: *Ad hoc On-Demand Distance Vector (AODV) Routing (RFC 3561)*. <http://tools.ietf.org/html/rfc3561>, July 2003. Status: EXPERIMENTAL.
- [121] PERKINS, C. E. und P. BHAGWAT: *Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers*. In: *Proceedings of the Conference on Communications Architectures, Protocols and Applications 1994 (SIGCOMM 1994)*, London, United Kingdom, August 1994.
- [122] PERKINS, C. E. und E. M. ROYER: *Ad-Hoc On-Demand Distance Vector Routing*. In: *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA 1999)*, S. 90–100, New Orleans, USA, February 1999.
- [123] PIÓRKOWSKI, M., M. RAYA, A. L. LUGO, P. PAPADIMITRATOS, M. GROSSGLAUSSER und J.-P. HUBAUX: *TranS: realistic joint traffic and network simulator for VANETs*. *Sigmobile Mobile Computing and Communications Review*, 12:31–33, January 2008.

- 
- [124] POSTEL, J.: *User Datagram Protocol (RFC 768)*. <http://tools.ietf.org/html/rfc768>, August 1980. Status: STANDARD.
- [125] POSTEL, J.: *Transmission Control Protocol (RFC 793)*. <http://tools.ietf.org/html/rfc793>, September 1981. Status: STANDARD.
- [126] RAMAKRISHNAN, M. und S. SHANMUGAVEL: *FPGA Implementation of AODV Routing Protocol in MANET*. In: *Proceedings of the 1st International Conference on Industrial and Information Systems*, S. 470–473, August 2006.
- [127] RAPPAPORT, T. S.: *Wireless Communications - Principle and Practice*. Prentice Hall Communications Engineering and Emerging Technologies Series. Prentice Hall PTR, Upper Saddle River, NJ 07458, 1996.
- [128] ROYER, E., P. MELLIAR-SMITH und L. MOSER: *An Analysis of the Optimum Node Density for ad hoc Mobile Networks*. In: *Proceedings of the IEEE International Conference on Communications 2001 (ICC 2001)*, Bd. 3, S. 857–861 vol.3, 2001.
- [129] ROYER, E. M. und C. E. PERKINS: *An Implementation Study of the AODV Routing Protocol*. In: *Proceedings of IEEE Wireless Communications and Networking Conference 2000 (WCNC 2000)*, Chicago, USA, September 2000.
- [130] RUHRUP, S., H. KALOSHA, A. NAYAK und I. STOJMENOVIC: *Message-Efficient Beaconless Georouting With Guaranteed Delivery in Wireless Sensor, Ad Hoc, and Actuator Networks*. *IEEE/ACM Transactions on Networking (ToN)*, 18(1):95–108, February 2010.
- [131] RUHRUP, S. und I. STOJMENOVIC: *Contention-based georouting with guaranteed delivery, minimal communication overhead, and shorter paths in wireless sensor networks*. In: *Proceedings of the IEEE International Symposium on Parallel Distributed Processing 2010 (IPDPS 2010)*, S. 1–9, April 2010.
- [132] SALIM, J., H. KHOSRAVI, A. KLEEN und A. KUZNETSOV: *Linux Netlink as an IP Services Protocol (RFC 3549)*. <http://tools.ietf.org/html/rfc3549>, July 2003. Status: INFORMATIONAL.
- [133] SCHEUERMANN, B., C. LOCHERT, J. RYBICKI und M. MAUVE: *A Fundamental Scalability Criterion for Data Aggregation in VANETs*. In: *Proceedings of the 15th ACM SIGMOBILE International Conference on Mobile Computing and Networking (MobiCom 2009)*, S. 285–296, September 2009.
- [134] SCHEUERMANN, B. und M. MAUVE: *Near-Optimal Compression of Probabilistic Counting Sketches for Networking Applications*. In: *Proceedings of the 4th ACM SIGACT-SIGOPS International Workshop on Foundation of Mobile Computing (Dial M-POMC 2007)*, August 2007.
- [135] SCHNAUFER, S. und W. EFFELSBERG: *Position-Based Unicast Routing for City Scenarios*. In: *Proceedings of the 9th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2008)*, Newport Beach, USA, June 2008.



- [136] SCHNAUFER, S., H. FÜSSLER, M. TRANSIER und W. EFFELSBERG: *SLOPE: A System for rapid Deployment of VANET Communication Protocols (Poster)*. In: *Proceedings of the 8th Annual International Conference on Mobile Systems, Applications and Services (MobiSys 2006)*, Uppsala, Sweden, June 2006.
- [137] SCHNAUFER, S., H. FÜSSLER, M. TRANSIER und W. EFFELSBERG: *Vehicular Ad-Hoc Networks: Single-Hop Broadcast is not enough*. In: *Proceedings of the 3rd International Workshop on Intelligent Transportation (WIT 2006)*, S. 49–54, Hamburg, Germany, February 2006.
- [138] SCHNAUFER, S., H. FÜSSLER, M. TRANSIER und W. EFFELSBERG: *Unicast Ad-Hoc Routing in Vehicular City Scenarios*. Techn. Ber. TR-2007-012, Department for Mathematics and Computer Science, University of Mannheim, December 2007.
- [139] SCHNAUFER, S., T. HAENSELMANN und W. EFFELSBERG: *BSMX: Message Exchange Between Unassociated WLAN Devices (Demo)*. In: *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems (SenSys 2010)*, Zurich, Switzerland, November 2010.
- [140] SCHNAUFER, S., J. KIESS, H. LEMELSON, S. KOPF, S. GEIGER und W. EFFELSBERG: *Proximity Detection and Message Exchange in WLANs*. In: *Proceedings of the 11th IEEE International Symposium on Network Computing and Applications (NCA 2012)*, S. 276–280, August 2012.
- [141] SCHNAUFER, S., S. KOPF und W. EFFELSBERG: *BSMX - A Prototype Implementation for Distributed Aggregation of Sensor Data*. In: *Proceedings of PhoneSense 2010 (SenSys 2010 Workshop)*, Zurich, Switzerland, November 2010.
- [142] SCHNAUFER, S., S. KOPF, H. LEMELSON und W. EFFELSBERG: *Beacon-based Short Message Exchange in an Inner City Environment*. In: *Proceedings of the 9th IFIP Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net 2010)*, June 2010.
- [143] SCHNAUFER, S., M. TRANSIER und W. EFFELSBERG: *PBM - Positionsbasierter Multicast für mobile Ad-Hoc-Netze*. In: *Basissoftware für drahtlose Ad-hoc- und Sensornetze*. KIT Scientific Publishing, March 2009.
- [144] SHANNON, C. E.: *A mathematical theory of Communication*. The Bell System Technical Journal, 27:379–423, July 1948.
- [145] SHARAF, M. A., J. BEAVER, A. LABRINIDIS und P. K. CHRYSANTHIS: *TiNA: a scheme for temporal coherency-aware in-network aggregation*. In: *Proceedings of the 3rd ACM International Workshop on Data Engineering for Wireless and Mobile Access (MobiDe 2003)*, S. 69–76, New York, USA, 2003.
- [146] SOMMER, C., Z. YAO, R. GERMAN und F. DRESSLER: *Simulating the Influence of IVC on Road Traffic using Bidirectionally Coupled Simulators*. In: *Proceedings of the IEEE Workshop on Mobile Networking for Vehicular Environments 2008 (MOVE 2008)*, S. 1–6, Phoenix, USA, April 2008.

- [147] STOJMENOVIC, I. und X. LIN: *Loop-free hybrid single-path/flooding routing algorithms with guaranteed delivery for wireless networks*. IEEE Transactions on Parallel and Distributed Systems, 12(10):1023 –1032, October 2001.
- [148] STOJMENOVIC, I. und X. LIN: *Power-aware localized routing in wireless networks*. IEEE Transactions on Parallel and Distributed Systems, 12(11):1122 –1133, November 2001.
- [149] TAKAGI, H. und L. KLEINROCK: *Optimal Transmission Ranges for Randomly Distributed Packet Radio Terminals*. IEEE Transactions on Communications, 32(3):246–257, 1984.
- [150] THORSTENSEN, B., T. SYVERSEN, T.-A. BJØRNVOLD und T. WALSETH: *Electronic shepherd - a low-cost, low-bandwidth, wireless network system*. In: *Proceedings of the 2nd International Conference on Mobile Systems, Applications, and Services (MobiSys 2004)*, S. 245–255, New York, USA, 2004.
- [151] TORRENT-MORENO, M., S. CORROY, F. SCHMIDT-EISENLOHR und H. HARTENSTEIN: *IEEE 802.11-based one-hop broadcast communications: understanding transmission success and failure under different radio propagation environments*. In: *Proceedings of the 9th ACM International Symposium on Modeling Analysis and Simulation of Wireless and Mobile Systems (MSWiM 2006)*, MSWiM '06, S. 68–77, New York, USA, 2006.
- [152] TOUSSAINT, G.: *The relative neighborhood graph of a finite planar set*. Pattern Recognition, 12(4):261–268, 1980.
- [153] TRIEBEL, T., S. SCHNAUFER und W. EFFELSBERG: *Achievement Notification in a Real-World Achievement System (Demo)*. In: *Proceedings of the 11th IEEE International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM 2010)*, WOWMOM '10, S. 1–2, Washington, USA, 2010.
- [154] TRIEBEL, T., S. SCHNAUFER, B. GUTHIER, H. LEMELSON, G. SCHIELE und W. EFFELSBERG: *REWARD a Real World Achievement and Record Database (Demo)*. In: *Proceedings of the 8th IEEE International Conference on Pervasive Computing and Communications (PERCOM 2010)*, S. 835 –837, April 2010.
- [155] TUDUCE, C. und T. GROSS: *A mobility model based on WLAN traces and its validation*. In: *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2005)*, Bd. 1, S. 664 – 674 vol. 1, March 2005.
- [156] U.S. DEPARTMENT OF COMMERCE: *Secure Hash Standard (SHS)*. National Institute of Standards and Technology, Washington, August 2002. Federal Information Processing Standard 180-2.
- [157] WEGENER, A., M. PIÓRKOWSKI, M. RAYA, H. HELLBRÜCK, S. FISCHER und J.-P. HUBAUX: *TraCI: an interface for coupling road traffic and network simulators*. In: *Proceedings of the 11th Communications and Networking Simulation Symposium (SCS SSM 2008)*, CNS '08, S. 155–163, New York, USA, 2008.

- [158] WERNER-ALLEN, G., J. JOHNSON, M. RUIZ, J. LEES und M. WELSH: *Monitoring volcanic eruptions with a wireless sensor network*. Proceedings of the 2nd Second European Workshop on Wireless Sensor Networks, S. 108–120, February 2005.
- [159] WILLIAMS, B. und T. CAMP: *Comparison of broadcasting techniques for mobile ad hoc networks*. In: *Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking & Computing*, MobiHoc '02, S. 194–205, New York, USA, 2002.
- [160] WISCHHOF, L., A. EBNER und H. ROHLING: *Information dissemination in self-organizing intervehicle networks*. IEEE Transactions on Intelligent Transportation Systems, 6(1):90–101, March 2005.
- [161] WISCHHOF, L., A. EBNER und H. ROHLING: *Self-Generated Road Status Maps based on Vehicular Ad Hoc Communication*. In: *Proceedings of the 3rd International Workshop on Intelligent Transportation (WIT 2006)*, Hamburg, Germany, March 2006.
- [162] WISCHHOF, L., A. EBNER, H. ROHLING, M. LOTT und R. HALFMANN: *SOTIS - a self-organizing traffic information system*. Proceedings of the 57th IEEE Semiannual Vehicular Technology Conference (VTC 2003-Spring), S. 2442–2446 vol.4, April 2003.
- [163] WITTEN, I. H., R. M. NEAL und J. G. CLEARY: *Arithmetic coding for data compression*. Communications of the ACM, 30(6):520–540, 1987.
- [164] XU, H. und M. BARTH: *Travel Time Estimation Techniques for Traffic Information Systems Based on Intervehicle Communications*. Transportation Research Record: Journal of the Transportation Research Board, Volume 1944:72–81, January 2006.
- [165] YAO, Y. und J. GEHRKE: *The cougar approach to in-network query processing in sensor networks*. SIGMOD Record, 31(3):9–18, 2002.
- [166] YOON, J., M. LIU und B. NOBLE: *Random Waypoint Considered Harmful*. In: *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2003)*, San Francisco, USA, March 2003.